1

AD-A241 114

INSTITUTE FOR SIMULATION AND TRAINING

Contract Number N61339-89-C-0043
PM TRADE
DARPA

January 15-17, 1990

Summary Report
The Second Conference on Standards for the Interoperability of Defense Simulations
Volume III: Position Papers

IST

Institute for Simulation and Training
12424 Research Parkway, Suite 300
Orlando FL 32826

91-11425

University of Central Florida
Division of Sponsored Research

IST-CF-90-01

91 9 24 080

INSTITUTE FOR SIMULATION AND TRAINING

Contract Number N61339-89-C-0043
PM TRADE
DARPA
January 15-17, 1990

Summary Report
The Second Conference on Standards for the Interoperability of Defense Simulations
Volume III: Position Papers

Editors:
Karen Danisas
Bob Glasgow
Brian Goldiez
Bruce Mcdonald

A-1

This report is informational and does not express the opinions of PM TRADE or DARPA.

Institute for Simulation and Training
12424 Research Parkway, Suite 300
Orlando FL 32826

University of Central Florida
Division of Sponsored Research

IST-CF-90-1

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS None
2a. SECURITY CLASSIFICATION AUTHORITY N/A 2b. DECLASSIFICATION / DOWNGRADING SCHEDULE N/A	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for Public Release; Distribution is Unlimited
4. PERFORMING ORGANIZATION REPORT NUMBER(S) IST-CF-90-1	5. MONITORING ORGANIZATION REPORT NUMBER(S) IST-CF-90-1
6a. NAME OF PERFORMING ORGANIZATION Institute for Simulation and Training 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Project Manager for Training Devices
6c. ADDRESS (City, State, and ZIP Code) 12424 Research Parkway, Suite 300 Orlando, FL 32826	7b. ADDRESS (City, State, and ZIP Code) 12350 Research Parkway Orlando, FL 32826
8a. NAME OF FUNDING / SPONSORING ORGANIZATION DARPA/TTO 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER NC1339-89-C-0043
8c. ADDRESS (City, State, and ZIP Code) 1400 Wilson Blvd. Arlington, VA 22209	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
Summary Report: The Second Conference on Standards for the Interoperability of Defense Simulations Volume III Position Papers

12. PERSONAL AUTHOR(S)
(Edited by) Danisas, Karen; Glasgow, Bob; Goldiez, Brian; McDonald, Bruce

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Summary	FROM ____ TO ____	1990, January 15-17	316

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Standards, Interoperability of Defense Simulations, Terrain Databases, Protocols, SIMNET

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This report presents a summary of the activities of the Second Conference of Standards for the Interoperability of Defense Simulations sponsored by the Defense Advanced Research Projects Agency (DARPA) and the Program Manager for Training Devices (PM TRADE). The workshop was hosted by the Institute for Simulation and Training/University of Central Florida (IST/UCF) on 15-17 January 1990. This volume contains a copy of all position papers received by IST/UCF by 15 February, 1990.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a. NAME OF RESPONSIBLE INDIVIDUAL Mike Garnsey	22b. TELEPHONE (Include Area Code) 407-380-4816	22c. OFFICE SYMBOL AMC PM-TND-EN

DD Form 1473, JUN 86 Previous editions are obsolete. SECURITY CLASSIFICATION OF THIS PAGE

SUMMARY REPORT

The Second Conference on Standards for the Interoperability of Defense Simulations

January 15-17, 1990 Orlando, FL

INTRODUCTION

This report presents a summary of the activities of the Second Conference on Standards for the Interoperability of Defense Simulations sponsored by the Defense Advanced Research Projects Agency (DARPA) and the Program Manager for Training Devices (PM TRADE). The workshop was hosted by the Institute for Simulation and Training / University of Central Florida (IST/UCF) on 15-17 January 1990, in Kissimmee, Florida.

This is the second workshop concerning the development of technical standards for networking defense simulations. These standards are intended to meet the needs of large scale simulated engagements systems which are being used increasingly to support system acquisition, test and evaluation, tactical warfare simulation and training in DoD. The primary goals of this workshop were to provide a forum and discuss issues prior to the development of a Protocol Data Unit level standard, to capture networking requirements and needs, and to exchange ideas and keep interested parties informed on networking technology issues.

The three day workshop focused on two major topic areas: Communication Protocols and Terrain Databases. The Communication Protocols was headed up by Dr. Ron Hofer, Chief, Engineering, PM TRADE. This group was mainly concerned with what goes over the wire. The following subgroups dealt with those issues:

* Interface
* Time/Mission Critical
* Security
* Long Haul/Wide Band
* Non visual

The Terrain Databases Working Group was headed up by George Lukes, Director of the Center for Autonomous Technologies, U. S. Army Engineer Topographic Laboratory. This group was mainly concerned with how the terrain data is interpreted. The following subgroups dealt with those issues:

* Correlation
* Dynamic Terrain
* Unmanned Forces
* Interim Terrain Database

In response to comments made at this workshop, a new subgroup is being formed to address Human Performance Measures. This subgroup will address requirements for recording and assessing student performance in the simulators on the network. As part of this effort, issues concerning instructor interfaces for controlling exercises and evaluating student performance will be addressed. User inputs about needed capability for networked simulators will be solicited. Mr. Bruce McDonald, Institute for Simulation and Training, will chair this subgroup, and any comments and suggestions should be directed to him.

This report has been separated into three volumes. Volume I contains summaries of all presenters' speeches. Volume II contains an attendees list, a copy of the view graphs used during presentations, and a copy of all documents that were submitted at the conference for the attendee information. Volume III contains a copy of all position papers received by IST/UCF by 15 February, 1990.

POSITION PAPERS

submitted for the

SECOND WORKSHOP ON STANDARDS FOR INTEROPERABILITY OF DEFENSE SIMULATIONS

POSITION PAPERS

Position paper: On adopting the Simnet Local Area Network Protocol as the Local Area Network Standard

In general we feel the Simnet local area network protocol is a viable candidate for the military local area network standard. It is both flexible and expandable, and distributes the processing needs well. There are however, at least two specific issues which must be addressed before the Simnet protocol can be applicable to the general military needs. These issues are:

1) Sending a matrix definition for the orientation (heading, pitch and roll) of a vehicle in the vehicle appearance PDU.

2) Sending non-dynamic and non-compact data in the vehicle appearance PDU.

We will now look at each of these two issues in detail, and then mention a few other concerns we have about the proposed protocol.

1. *Sending a matrix definition for the orientation (heading, pitch and roll) of a vehicle in the vehicle appearance PDU.*

We believe this is the most serious issue in keeping the Simnet protocol from becoming applicable to the general military needs. By sending a matrix for the orientation of the vehicle instead of the actual heading, pitch and roll angles, two important abilities are lost. First, the ability to extrapolate (or dead-reckon) the vehicles orientation is lost, since in matrix format the actual angles are not extractable, and extrapolation with the matrix is not viable. This is probably not critical in a tank simulation, where changes in the orientation of a vehicle is generally slow. In a high fidelity aircraft simulator (where heading, pitch and roll changes happen both rapidly and sporadically) however, the ability to extrapolate is paramount. Without this ability, the network load to keep the visual image acceptable is astronomical. The ability to extrapolate the orientation by higher order equations is also lost (e.g., an airplane in a smooth banked curve).

The second problem is the inability to time correct for network delays. Since the simulator does not have the actual angles, it can not determine the proportion of correction needed based on the time variation. Time corrected data will be very important in a network collision environment where the time to transfer a packet is indeterminate.

We believe there are definite advantages to sending a matrix to describe the orientation of a vehicle, but we believe these advantages are outweighed by the abilities that are lost (extrapolation and time correction) in sending a matrix instead of the actual angles. These two abilities are fundamental to a high fidelity simulation. We also believe that the extra 24 octets (20% of the vehicle appearance PDU) needed to describe the matrix as apposed to the actual angles is also more detrimental to the system than the extra processing needed to recompute the orientation matrix for each vehicle in the field of view. We believe the network traffic is going to be the limiting factor in simulation fidelity and expandability, and reducing the amount of information on the network should be first priority.

2. *Sending Non-dynamic and Non-compact data in the Vehicle Appearance PDU.*

As stated previously, we believe that network traffic will be the limiting factor for simulation fidelity and expandability, and should be addressed with the highest priority. In the July 31, 1989 "THE SIMNET NETWORK AND PROTOCOLS" manual by BBN Systems and Technologies Corporation, the vehicle appearance PDU, which will comprise most of the activity on the local area network, has the following non-changing fields in it.

1) vehicle class	(1 octet)	4) markings	(12 octets)
2) force	(1 octet)	5) capabilities	(4 octets)
3) guises	(8 octet)		

This is a total of 26 octets out of 120 (21.7%), which will not change, and are not needed except at initialization (activation) of the vehicle or when a new vehicle is added to the simulation. All this information about other vehicles could be stored relative to the vehicle's ID. Only when a vehicle is added to the simulation should each vehicle broadcast the above information about itself, allowing the added vehicle access to this static information. Other inefficiencies in the vehicle appearance PDU are the following. Sixty-four bit floating point numbers to represent the vehicle in world coordinates is overkill. Forty bits is enough accuracy to position a vehicle anywhere on the world with an accuracy of above 1/100 of a foot. We realize that a 5 octet number (40 bits) is not as convenient as a 64 bit number but an easy conversion could be made after the number arrives at the simulator. This change would save 9 octets (7.5%). Also, it seems reasonable to compact engine speed into 1 octet and the stationary flag into one octet, saving 2 more octets (1.7%). Combining these changes with sending the actual angles instead of a matrix (a savings of 24 octets), 61 octets (51%) of the total size (120) of the vehicle appearance PDU can be saved. Since the vehicle appearance PDU will comprise the majority of the traffic on the network, a substantial increase in performance and expandability should be attained.

Large packet sizes not only decreases the performance and expandability of the network, but also increases the processing time for each individual simulator to read the additional information off the network.

3. *Other concerns about the proposed standard.*

1) For high fidelity simulations, time stamping to an accuracy of a millisecond is too coarse. We suggest an accuracy of at least 1/8 of a millisecond.

2) A protocol for communicating changes in the state of the database (blown-up buildings or bridges) to vehicles entering the simulation late has not been proposed. We believe this is a critical issue, which must be addressed early in the design.

3) A constant/near constant delay is impossible to attain in a non collision free network environment, since wait periods are pseudo randomly generated after a collision. This will be a problem for high fidelity simulations such as aircraft dog fighting. We believe a protocol change might be needed to allow for a more predictable network delay in certain situations.

4. *Conclusion*

The Simnet protocol is well suited for the slow moving simulations, such as tank simulators. It is not, in it present state, however, applicable to high fidelity, fast moving simulation such as aircraft dog fighting. Since these types of simulations will most likely be part of the future military network, modifications to the proposed standard are needed to make it applicable to these cases. In particular, the following changes need to be made:

1) The actual orientation angles need to be sent in the vehicle appearance PDU.

2) All PDUs should be as compact and small as is practical. Especially the vehicle appearance PDU, since it will comprise the majority of the network traffic.

3) The following issues as they relate to high fidelity, fast moving simulations, should be addressed as soon as possible: time stamping accuracy, notification of state changes in the database, and variations in network delay.

Time/Mission Critical Issues
For Networks of Simulators

Gary R. George
Staff Engineer
CAE-Link Flight Simulation

An Issue Paper Prepared
for the Second Workshop on
Standards for Interoperability of
Defense Simulations Kissimmee, FL 15-17 January 1990

Introduction

Adverse effects of aviation simulator latency and low computer update rates of single simulation devices have been well documented (1-5)*. Also an FAA standard has been developed to specify latency in single simulators for various training levels (6). The effects of latencies on simulator trainees has been the inability to perform high gain tasks such as aerial refueling and a phenomenon known to simulator trainees as simulator sickness.

Published research into the effects of latencies in regard to networks of simulators has been limited (7,8). As with the case of a single simulator it is expected that latencies in data transfers between devices will have some adverse or undesirable effects. Johns (8) has found from preliminary research that 300 ms is the maximum delay tolerable for certain fighter air to air weapon engagements in networked simulators. Although this issue paper will not attempt to set a minimum network latency or update rate (since there is insufficient research to base it on) there are a number of specific examples of time or mission critical tasks which will most likely or from our experience in actual simulator networking be very critical in regards to team training and mission rehearsal.

* This is only a partial list of references (For further references please contact the author)

Major Issues

The aviator like other members of the combined arms team needs to process a great deal of information, but the aviator's operating environment has an added dimension, height above the terrain. This and his mobility on the battlefield, force him to learn to perform with three distinct differences over other ground based elements:

1) The amount of information the aviator has to process is larger due to aircraft complexity, and his larger picture of a complex battlefield environment.

2) The processing of that information must be accomplished quickly due to the high mobility of the aircraft and its threats.

3) The accuracy of the aviator's tactical decision making must be correct the first time, even under the time pressures typical of air combat.

This means high workloads under stressful conditions, near information overload with time as a critical variable. An error in decision making or situational awareness can result quickly in catastrophe. This is not meant to imply that a decision-making error in ground based forces, for example, cannot have a tragic end. The important point is that the aviator tends to have less time to correct an error as well as having another dimension, and another time frame within which to make it.

In order to provide realistic team training and mission rehearsal with these high speed workloads for the aviator via networks of simulators the effects of latency and update rates between devices is a critical issue for various time compressed tasks.

The following discussion will use four specific examples to illustrate the adverse effects on mission critical tasks due to latencies and low update rates. These examples from our experience are only a few and as further networking research is done other mission critical tasks for the aviator in particular will be added. The mission/time critical examples include:

1) Escort flight with several aircraft including different types
2) Air-to-air refueling involving several aircraft
3) Air-to-air combat
4) Coordinated attack such as target handover

For each, the considerations of both update rates and overall latency will be discussed.

Background

Much of the contents of this report are from actual networking experience of Multiple Simulator Networking (MULTISIM) by CAE Link (9-15).

Expansion of Issues

Escort Flight - Military operations will use teams of rotor wing and fixed wing aircraft in support of ground and naval forces. Rotor wing aircraft will be combined into teams of attack and scout aircraft and will require teams to land in confined landing zones. Others will be used as cover for unarmed utility aircraft. Fixed wing aircraft will fly with various wingmen. This escort flight will be done at high speed with aircraft in close proximity of each other. For rotor wing aircraft this will be done at nap-of-the-earth using the terrain and cultural

features to the pilot's advantage. In actual operation with night conditions as a further disadvantage several tragic mid air collisions have occurred including Desert 1. In order to effectively train team escort flight it is essential that aircraft position of networked devices re`ative to each other be precise otherwise false crash cues can be expected.

A further analysis can be done if the crash mechanism from simulation is examined. The crash response normally comes from the simulator image generator when there is a intersection of crash volumes or ownship faces.

A typical crash volume is defined for a rotor wing aircraft in figure 1 and is typical of some image generators. It consists of volumes represented by polygons around the ownship model and the rotors in the case of the rotor wing aircraft shown. Crash indication between aircraft would then occur when volumes between simulators on the network intersect. Some intersections which just barely intersect the volume might result in a soft crash indication such as a slight bump cue perhaps. Massive intersections would, of course, result in a catastrophic crash conditions with resulting cues.

SIMNET protocol uses a dead reckoning technique which works very well for ground type simulations. The technique requires each device to maintain a detailed model of itself as well as an extrapolation model which all other network devices have also. When the error between these states for rectilinear position is more than 10% of the vehicle dimension or more than 3 degrees in attitude the accurate position from the detailed model is broadcast on the network. The end result is to optimize the network traffic and maintain it within the bandwidth of the system.

Consider the rotor wing aircraft model in figure one. This Chinook (UH47) is approximately 54 feet in length from rotor tip to rotor tip. A 10% error then corresponds to 5.4 feet for the longitudinal axis. The UH-47 has a cruising speed of 160 knots or 270 ft/sec. For the SIMNET protocol the maximum state update i.e. zero dead reckoning is 15hz. It is easy to develope the distance traveled in that time at 160 knots to be:

$$s = vt = 270(.067) = 18 \text{ feet}$$

This is already three times the threshold indicating the update rate to be insufficient. If a modern fighter with air speed much greater than UH-47 is analyzed the displacements increase much more. It is important to note that this does not include the latency due to the network medium itself further compounding the problem.

The granularity of the data then produces uncertainty in position both in the data base in regard to aircraft crash volumes or faces intersecting cultural features, terrain and to other networked devices. The result being that false crash indications are to be expected for networked aircraft working in close confined escort missions. The immediate effect for team training for the user would be to fly further apart and keep maneuvers simple unlike that found in actual combat.

A further consequence of this 15hz update and network delays will be large steps in state resulting in stepping or jitter of visual presentations of networked simulators resulting in eye strain and possible disorientation or simulator sickness.

Air-to-Air Refueling

This very difficult task is necessary in a number of missions including mission rehearsal. Figure 2 shows a typical rotor wing

refueling. It involves approaching the fueler and docking the probe and basket and then maintaining coordinated flight with both the refueler and possibly another refueling aircraft. Effects of turbulence require exacting aircraft controls - visual coordination between the aircraft, probe and basket.

Currently simulation which is used for aerial refueling training for single devices includes an automated refueler for a single simulator. Aerodynamic and turbulence modeling as well as crash conditions for probe and basket (bump cues for probe/basket intersection) are considered. Therefore over the network comes information about not only the aircraft state but the position of the hose and basket relative to the refueler aircraft. In the content of networking this extremely high gain task (and in many cases a mission critical task) can only be done with sufficiently high update rates of state between devices. We have found in single simulators with automated refuelers that even aircraft updates at 30hz can cause pilot induced oscillations in docking the probe and basket. In trying to dock with the basket, the basket as it moves must be smooth in the visual presentation and its position relative to the probe must be precise. With the turbulence the boom will move around significantly making it jitter for dead reckoning or extrapolation. Deadbanding these effects should not be considered since the effects of small displacements are important for training.

If the state updates between devices are inadequate the ability to perform air-to-air refueling will be significantly impaired or impossible to perform in a team training setting.

Air-to-Air Combat

This high gain task requires that aircraft be flown at top performance using the total aircraft and system to defeat a foe in aerial combat. There are several combinations such as one on one, two on one, etc.

Each member of a networked simulation system is faced with both opponents and defenders whose present position and orientation is not current. The state is representative of some state along the flight path at a past time depending on the latency. For the highly dynamic nature of air-to-air combat differences in both range and orientation between devices will widely vary.

As previously discussed the uncertainty of position can also cause false crash cues for air-to-air combat. Another area where low update rates and time delays between networked device will cause problems is weapon scoring. Johns (8) has pointed out time delays will tend to be advantageous to the attacking aircraft. This is due to the fact that the defending pilot and his simulated sensor system would suffer delay in what is sensed of an attack and countermeasures (e.g. jamming, flares, etc.) would be delayed to counteract the attackers weapons. Air-to-air combat is split second decisions and actions to survive and fight again in the real world. Thus, the attacker has a false advantage in a team training setting and may score better in kills that in reality would not be that high.

Uncertainty in position also provides uncertainty in regard to where weapons hit and the ability to kill targets, further degrading realistic scoring. The amount of delay will depend on the weapon type with Johns (8) noting gun engagements in particular can tolerate a maximum 300 ms delay for the defender aircraft.

As with escort flight, jumping or jittering targets that would be expected with low update rates and large latencies will cause eyestrain and fatigue in air-to-air engagements.

<u>Coordinated Attack</u>

This type of team effort for aviation can include target handover between different aircraft. Examples include remote lasing of a target while another aircraft fires a laser guided missile and the coordinated effort between the A10 and AH64 in Joint Air Attack Team (JAAT) (17) with the Maverick missile.

Figure 3 shows a typical target handover. Aircraft A lases the moving ground target while Aircraft B fires a laser weapon. This is a fire and forget mode for aircraft B. The laser simulation for more advanced devices includes beam width effects, visibility and beam spill over. These effects are important for training of mission ready optimum lasing techniques that has been successfully implemented on the AH64 Combat Mission Simulator (18) for the training of mission ready Army crews. The exact position of the weapon hit is also important factor. Where the missile hits on the target will have an effect on the kill status. For example if a track is hit the target is immobilized. If it hits reactive armor then there may be little effect. If the missile trajectory is flown by the firing aircraft simulator it will fly to the perceived or delayed laser spot position as designated by aircraft A. The key for the operator of aircraft A for a kill is to have the laser centroid on the target at the terminal approach of the weapon. This includes lasing at the last minute such that detection or counter-measures are not used by the threat. Due to errors in target position (caused perhaps by inadequate update rates) or different delays between the target position and the laser spot position transmitted to

simulator B it is possible to have perfect lasing on target by simulator A but have simulator B score a miss since there is a miscorrelation between the laser position and where the missile actually hits the terrain in the world of B.

It should be apparent that this is a difficult correlation problem to solve for significant delays due to slow update rates and network latency. Since the delays for target position and laser spot position will be different, time tagging for dead reckoning or extrapolation of these states would be necessary. Also, extrapolation accents the noisiness of the laser spot position causing it to jump and jitter especially for high angular sensor rates common to current sensor systems (in excess of 60 degrees/sec) with auto trackers and manual inputs combined with long delays. Possible uses of modern estimation theory may be useful for this difficult problem.

A solution (at least partially) is to fly the weapon from the designating aircraft A. There is still delay between the target position and the designating aircraft for which dead reckoning or extrapolation must be used to predict target position relative to the laser spot. Again, the problem of noise and the laser spot jumping has to be considered and has been a considerable problem to solve from our experience. A further disadvantage to this approach is that each device which can be a designator must have complex weapon models.

Another approach has been to just compare the line-of-sights (LOS) of the targets with the designator LOS and assume that the LOS closest to the designation is the target to be scored or impacted. A roll of the dice then determines lethality of hit rather than a more exact hit method as previously described. There exists the potential with multiple targets to select the wrong one especially if the targets are close together.

Recommendations

Some specific examples of time/mission critical tasks have been presented. From our experience with networked aviation simulators especially for high gain tasks it is apparent that high update rates are required with minimal network latency to perform these functions. We know also that 15hz and low bandwidth network media work well for ground based selective fidelity devices.

The question then is really what is required for various types of simulators, in regard to latencies and update rates.

The following should be considered:

1) A combined effort of industry, academia and government to analyze various team tasks in regard to specific simulators and determine latency and update rate requirements. The extensive research done for single simulator latencies can become a foundation for this research. Also, there is a considerable amount of expertise in simulator latency (NASA, NTSC, ASD, industry, etc.) that could be used in this study.

   The bandwidth of the network must be determined by task and mission requirements, much like the concept of mission oriented simulator design for simulator development recently applied to several single simulators. (19,20). Latencies and low update rates can cause anomalies and systems characteristics that are not representative of actual operations in team training. This is extremely important for team training as well as weapon systems evaluations that has been promoted recently using networks of simulators

where these effects could bias criteria for an actual weapons or avionics development.

2) The concept of groups of low and high bandwidth networks interacting via smart gateways i.e. being interoperatable must be considered.

References

1. Kennedy, R.S., Dutton, B. Richard G.L., Frank, C.H., "Simulator Sickness: A Survey of Flight Simulators for the Navy," SEA 841597, 1984

2. Ullano, K.C. Lambert, E.Y., Kennedy, R.S., Sheppard D.J., "The Effects of Asynchronous Visual Delays on Simulator Flight Performance and the Development of Simulator Sickness Symptomatology," NAVTRASYSCEN 86-D-0026-1, 1986.

3. VanHoy, B.W., Allgood, G.O., Lillenthal, M.G. Kennedy, R.S. Hooper, J.M., "Inertial Control Systems Measurements of Two Motion-Based Flight Simulator Sickness," Image IV Conference, Phoenix, Arizona.

4. Lillenthal, M.G., Kennedy, R.S., Hooper, J., "Vision Motion Induced Sickness in Navy Flight Simulators: Guidelines", I/ITSC, 1987.

5. Niemeyer, G.E., "Measurement of Flight Simulator Time Delays," AIAA Flight Simulation Conference, 1987.

6. AC-121-14C Advisory Circular; Department of Transportation, Federal Aviation Administration, 8/29/80.

7. Malone, Hugh L.; Horowitz, Scott; Brundeman, John A.; Eulenback, Hubert; "The Impact of Network Delay on Two-Ship-Air-to-Air Combat Simulation"; AIAA Flight Simulation Conference, 1987.

8. Johns, L; "A Study of the Effects of Delay Times in a Dome-to-Dome Simulation Link"; AIAA Flight Simulation Conference, 1988.

9. George, G.R., Knight, S.N., and Monette, R.; "Multiple Simulator Networking (MULTISIM) - The Way to Provide Effective Combat Training Today", I/ITSC, Orlando, FL., 1988.

10. George, G.R., Knight, S.N., and McTiernan, J.; "Networking of Full-Fidelity Simulators for Advanced Mission Training"; NAECON, Dayton, OH 1989.

11. George, G.R., Knight, S.N., Stark, E.A.; "Fidelity Requirements for Aviator Training Networks" AIAA Flight Simulation Conference, 1989.

12. Demers, W.A.; "Son of Simnet"; MILITARY FOREM, October 1989.

13. Michelson, Chad; "US Army Simulation Training: No Longer A Game of Solitaire"; November, 1989.

14. Schneider, Ronald, "Simulation and Team Training"; Army Aviation; October 1989.

15. Monette, Bob, "MULTISIM"; United States Army Aviation Digest; November, 1988.

16. Thorpe, J.A.; Bloedom, G.W.; Hayes, M.F.; "The Simnet Network and Protocols", DARPA Report 7102, July 1989.

17. Ferrell, Mark; Reynolds, Scott; "Apache Thunder - Advanced JAAT Test"; Army Aviation, May 31, 1989.

18. George, G.R., Knight, S.N., Drew, E.W.; "AH64 Combat Mission Simulator Tactical System", AIAA Flight Simulation Conference, 1987.

19. George, G.R., Knight, S.N., Stark, E.A., "Mission-Oriented Simulator Development", AIAA Flight Simulation Conference, 1988.

20. Monette, R. Knight, S.N., George, G.R., "Achieving Mission Rehearsal Objectives Employing Mission-Oriented Simulator Development", I/ITSC, 1989.

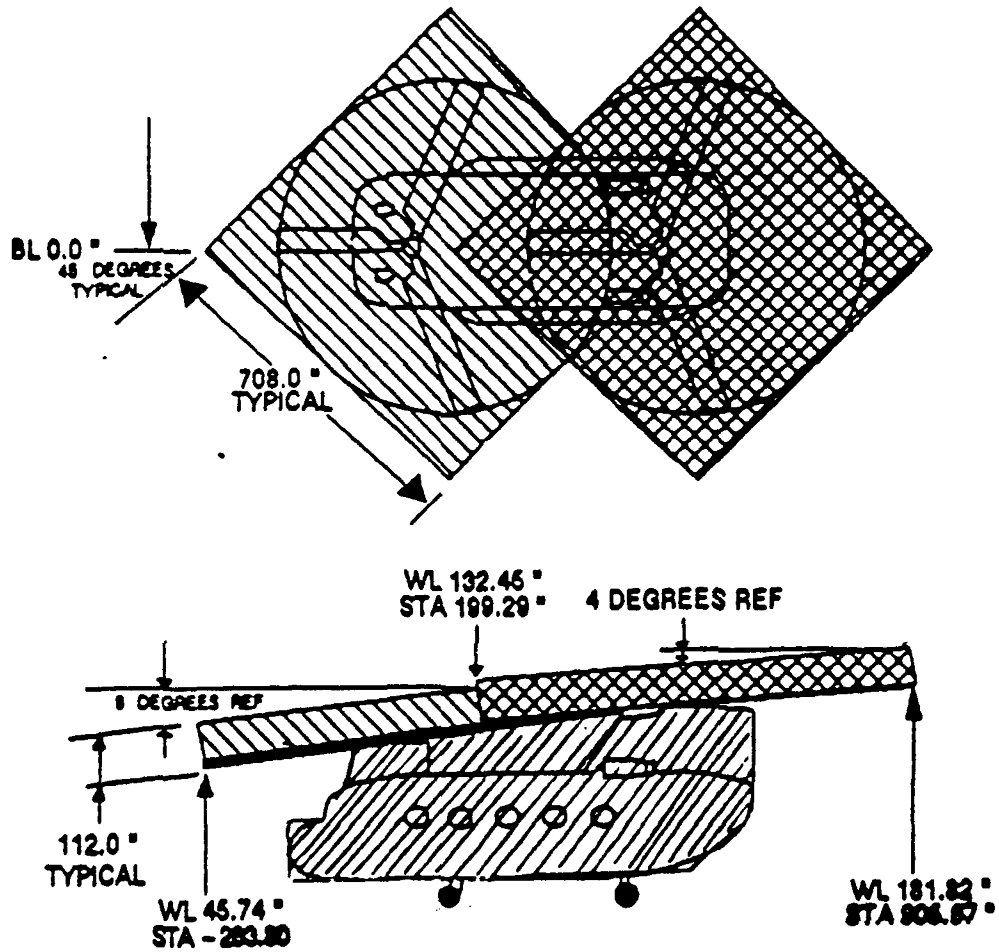


Figure 1 Crash Volumes

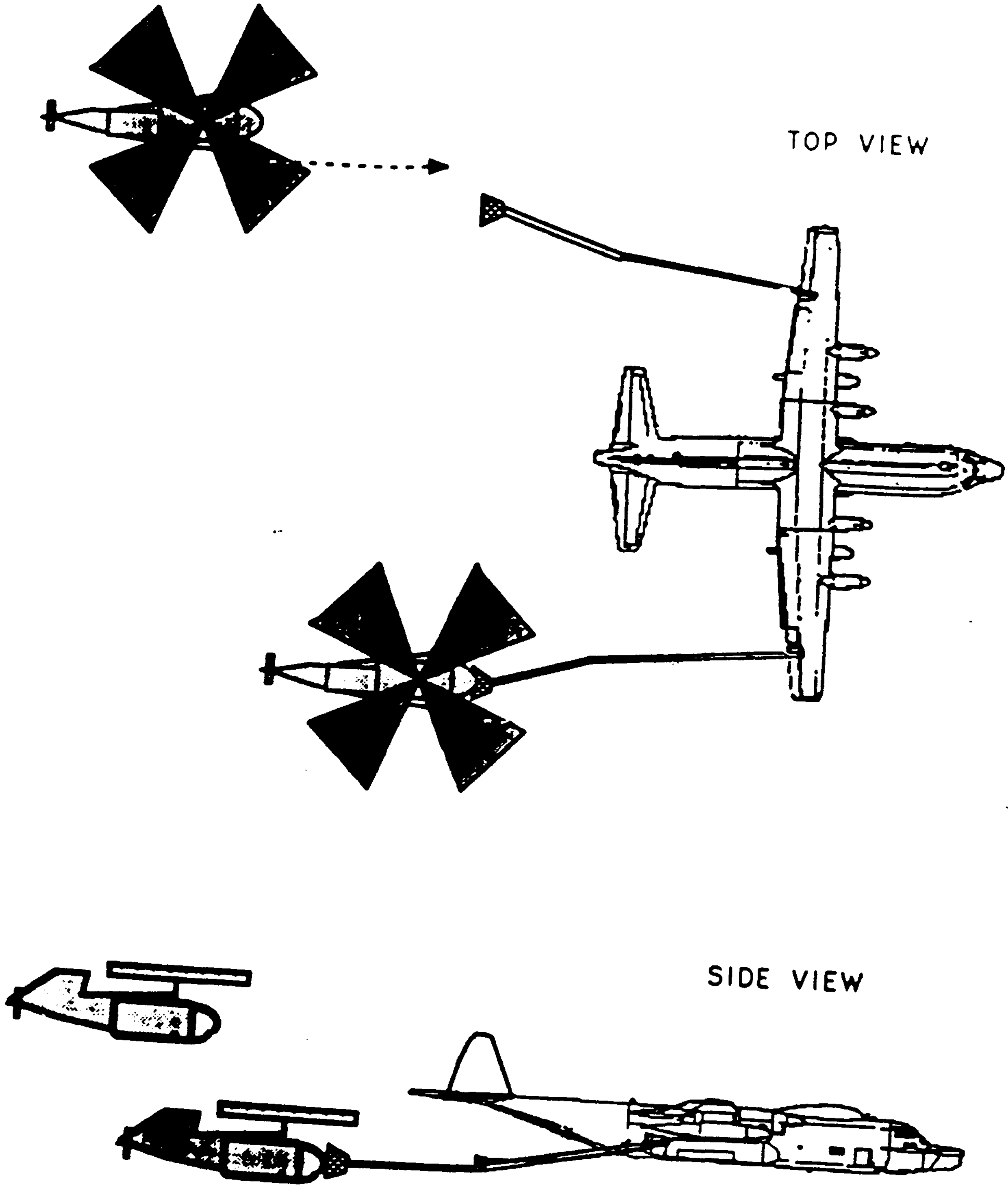


Figure 2 Air-to-Air Refueling

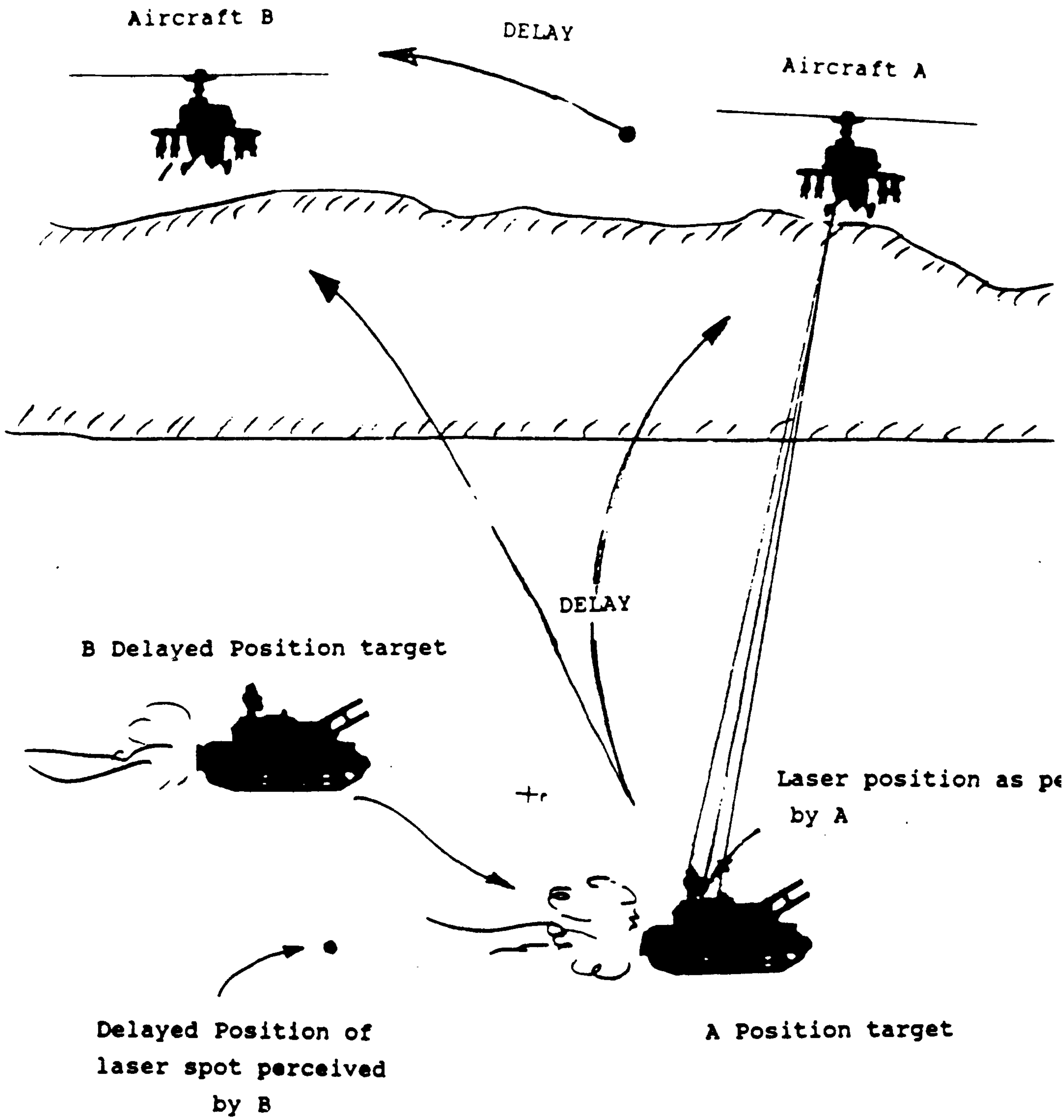


Figure 3 Target Handover

Environmental Correlation in Networks of Simulators

Gary R. George
Staff Engineer
CAE Link Flight Simulation

An Issue Paper Prepared
for the Second Workshop
on Standards for Interoperability
of Defense Simulations
Kissimmee, FLA 15-17 January 1990

INTRODUCTION

Military aviation forces are responsible for a number of specific missions which take advantage of the aviator's mobility and greater view of the battlefield. Due to the height degree of freedom the aviator is very concerned with his environment and how he can effectively use it to his advantage. For example, teams of rotor wing aircraft use the terrain to their advantage being able to accurately navigate to positions avoiding threats. For special operations crews, long range rendezvous missions involving several aircraft will be necessary under various weather conditions. Close air support teams for air to ground weapon delivery will require specific navigation to the forward area of battle to effectively avoid threats on their planned course.

As team training objectives are developed and the use of networked simulators for advanced mission training and rehearsal becomes available, the concerns over correlation of environmental simulations between networked devices to support mission training and rehearsal will be particularly important. This issue paper will define the issues in regard to environmental correlation for networked simulators of both existing and new devices. Furthermore, some options for each issue will be suggested for consideration.

Major Issues

The major issue topics of Environmental Correlation as they relate to networking are the following subjects:

1. Attributes of the Navigation Facilities
2. Earth Model Definition
3. Global Positioning System Satellite Coverage

4. Magnetic Variation Determination
5. Pressure, Wind, and Temperature Models
6. Weather/Smoke/Special Effects

Background

Much of the contents of this report are from actual networking experience of Multiple Simulator Networking (MULTISIM) by CAE Link (1-5). An over-all general review of navigation can be found in reference (6), Army FM1-5. Outside references are very limited in the general subject of environment simulation (7,8).

Expansion of Issues

Attributes of Navigation Facilities - Actual aircraft missions need a number of navigational facilities (TACAN, VOR, ILS, etc.) for navigation. Navigation Environment Simulations require the storage of radio/navigation and communications systems data.

This data must be easily accessible and useable in real time. This data, also, must be easily updated including addition or deletion of stations, as well as changing existing station data. Furthermore this data may need to be changed quickly for mission rehearsal.

In order to accomplish this, many simulations use a special compiler to build or modify a file of radio navigation facility parameters in disk storage. This data is obtained from the DoD Flight Information Publications (ref. 9). The data in disc storage is usually different for various simulators. Facility type data is dependent on the aircraft communications and navigation equipment and the mission of the simulated aircraft. This difference can obviously cause a problem with a number of networked simulators performing a coordinated mission.

Navigational aids should be common for the various devices. The solutions to the problem are:

1. Change the navigational facilities file such that it is common for each device. This would require a coordinated effort to review the data with a number of different contractors for each simulator. Also, facilities parameters are updated often and a continual coordinated update would be required. Retain separate and simulator unique data for individual training. Disadvantages are that file space may overflow and there is the added task of redefining the necessary navigational facility data.

2. Define unique navigational facility data in a similar manner to (1) but have it reside in a central point on the network with a unique node. Disadvantages in this approach include the requirement to change some indexing on each device for a common list and provide logic to use the central point data only in network mode.

Earth Model - In simulation, positions of the aircraft are derived as a function of the equations of motion. More advanced simulations use navigation geometry software to integrate the resultant aircraft velocities into latitude and longitude. Latitude and longitude is converted to UTM coordinates when required. This defines the actual position of the simulated aircraft in the world. Differences in the type of earth model (e.g. flat earth, spheroid, world geodetic system) used by each simulator could provide inconsistencies in the locations computed by networked simulators.

This is of particular concern for long distance interception and coordination of location between aircraft. Another consideration is that the particular earth model used will be dependent on the simulated environment (e.g. it may vary as a function of the location in the simulated world). Options available for the earth model include:

1. The use of a central point on the network to provide all devices with a uniform model. Use of a unique network node or central point on the network would require each device to provide rates from equations of motion which could be integrated into the corresponding latitude and longitude as determined by the specific earth model at the central point. For the non-network mode, each device would revert to its own model. A major disadvantage of this approach is transport delays for positioning update data.

2. Change earth modeling to a uniform one for all simulators. In many cases, the particular model is dependent on the on-board avionics system. It may be necessary to have the original earth simulation modeled after the on-board avionics and the uniform one for networked positioning. A general trend on earth model simulation is the World Geodetic System (WGS)-84 . Disadvantages to this approach include software changes at each simulator and definition of the most advantageous model.

Global Positioning System (GPS) Satellite Coverage - The use of the Global Positioning System for accurate navigation will be particularly important for deep strike penetration and special operations. The effectiveness of this system is the satellite coverage at the time of the mission. Four satellites are needed

to provide accurate three-dimensional position, as well as coordinated universal time, and velocity. Three satellites can provide slightly degraded operation. Obviously, missions will be defined such that coverage is optimal if possible. Certainly, other factors such as time of day, weather, etc. will be considered.

Currently, modeling of satellite coverage for GPS simulation is done on some simulators. Typically, the GPS position is derived from the simulated aircraft position on the earth model plus instructor induced GPS positional errors. For complex networked team missions, (particularly unique mission rehearsal), the effects of satellite coverage and its relationship to navigation accuracy is necessary. The effect of one team being delayed (malfunctions, lost time, etc.) in a support mission and loss of accurate GPS requiring use of other systems must be considered. The options for the GPS satellite simulation are:

1. Use detailed models somewhat like that available from Dr. Glenn Siebert from SRI International. Disadvantages here are that computer resources would be taxed. The most likely place for an advanced simulation similar to this is at a central point on the network.

2. The most likely approach is to use a file with satellite data to determine coverage and characteristics. This approach could be done at each simulator but would be best accomplished at a central point.

Magnetic Variation

Magnetic variation always seems to be a confusing subject. The difference between true and magnetic north varies at different earth locations. Differences in the way magnetic variation is defined between networked devices will cause navigational errors for intercept and coordinated attack. Simulation of magnetic variation in the navigational environment is done in a number of ways:

1. Provide constant value of magnetic variation for simulations which use relatively small data bases.

2. Use of a spherical harmonic model (similar to the one used to produce the Naval Oceanographic Magnetic Variation Charts) to generate magnetic variation for a given location, altitude, and date.

3. Interpolation of data using a very large data base magnetic variation values (in special files) for the entire planet or specific gaming area. The DoD publishes magnetic variation in formation for airport locations in Flight Information Publications (FLIPs) (9).

The options available to get magnetic variation consistent in networked simulators are: Use of a general method or model at each device or apply that same model at a central point on the network. A device with a unique node on the network would take each network device's position and determine the corresponding magnetic variation based on some model. Consideration of the team mission is necessary. Long range mission rehearsal certainly cannot use fixed or constant magnetic variation.

Pressure, Wind, and Temperature (PWT)

Pressure, wind, and temperature simulation provides a weather environment in which pressure pattern navigation can be accomplished and in which pressure, wind, and temperature have a realistic effect on aircraft system performance. Correlated PWT simulation for networked simulators is necessary for team training. Naturally, the simulation varies among simulators based primarily on mission and simulated global position including:

1. Constant values set at initial conditions or by instructor edits similar to existing simulators. Some may be altitude dependent such as wind, speed, and direction at different altitudes.

2. Provide disk data on wind speed, wind direction, outside air temperature, and the true altitude associated with each of the pressure layers at a grid point. Interpolation is done at positions away from the specific points.

Options to incorporate PWT into networking are similar to other navigational environment issues. That being to define a necessary model for the team training mission requirements and incorporate it at each device or at a central point on the network.

Effects of Weather

Pilots have historically made tactical decisions based on the weather situations. Weather effects especially over large bodies of water can change quickly and flight crews must be familiar with its possible consequence on mission performance. Little has

been published on weather effect requirements (10) for simulation.

For networks of simulators there are two issues:

1. The need for realistic weather simulations for long range mission rehearsal and team training requirements.

2. Consistent or correlated weather and visibility conditions (for out the window visual, sensors and weapons) between networked devices. Associated issues include special effects of smoke and fog on various laser and weapon performance for networked devices.

Recommendations

The navigation environment issues are much like that of the tactical threat environment. Many of the navigation functions of environment would be best controlled and environmental factors determined at a central point to be shared by all networked simulators. The position of each device would be sent over the network to the central point and the various environmental factors determined (e.g. magnetic variation) and sent back to the devices. The concept of a "Universal Environment Simulator" to include both tactics and navigation environments for all simulators on the network may be necessary.

Definition of the models (e.g., earth, magnetic variation) will be determined by the team training and overall mission objectives. A utilization study would be helpful in this determination. For the short term, the following needs to be accomplished:

1. Determine differences in accuracy between various earth models currently used in simulation.

2. Define an adequate satellite model or look up tables for GPS simulation. This should be done in context of mission rehearsal requirements.

3. Study the possibility of a Universal Environment Simulator on the network for tactical and navigation environment.

4. Determine weather simulations necessary for team training.

5. Determine visibility correlation for out the window, sensors and weapons.

REFERENCES

1. George, G.R., Knight, S.N., and Monette, R.; "Multiple Simulator Networking (MULTISIM) - The Way to Provide Effective Combat Training Today"; Interservice/Industry Training Systems Conference, Orlando, FL., 1988.

2. George, G.R., Knight, S.N., and McTiernan, J.; "Networking of Full-Fidelity Simulators for Advanced Mission Training"; NAECON, Dayton, OH 1989.

3. Demers, W.A.; "Son of Simmet"; Military Forem, October 1989.

4. Micheloon, Chad; "US Army Simulation Training: No Longer A Game of Solitaire"; November, 1989.

5. Schneider, Ronald; "Simulation and Team Training"; Army Aviation; October 1989.

6. FMI-5 Instrument Flying and Navigation for Army Aviators, July, 1989.

7. Stanzione, Thomas, "Navigation and Defensive Simulation in the U-2 Cockpit Trainer," NEACON 1983. Proceedings, IEEE, Dayton, Ohio.

8. Taylor, John W., "Dynamic Simulations - Going Beyond With Today's Technology," NEACON 1984 Proceedings, IEEE, Dayton, Ohio.

9. DoD Flight Information Publication, Defense Mapping Agency Aerospace Center; St. Louis, Missouri.

10. Hart, Bruce H., "Fleet Requirements for F-14D Aircrew Trainer Suite," AIAA Flight Simulation Conference, 1989.

NETWORKS/wp50/kma

Issues Affecting the Networking of Existing and Multifidelity Simulations

Samuel N. Knight
Staff Scientist
CAE Link Flight Simulation

An Issue Paper Prepared
for the Second Workshop
on Standards for Interoperability
of Defense Simulations
Kissimmee, FL 15-17 January 1990

Scope

This position paper addresses the networking issues of interfacing existing simulators and simulators of different fidelities.

Introduction

Link Flight Simulation has been actively involved in advanced real-time simulation networking for several years. Link's Multiple Simulation Networking (MULTISIM) team has concentrated its efforts on meeting the combined-arms training needs of the aviation community with emphasis on the demanding nap-of-the-earth operational requirements imposed by helicopter attack teams, including air-to-air scenarios. Through the MULTISIM program, Link has acquired tremendous experience in real-time networked system performance, techniques for interconnecting existing and dissimilar devices, and advanced concepts and technologies for networking future multi-fidelity systems. Close developmental coordination has been maintained with the user community including numerous demonstrations and pilot evaluations involving networks of multiple, dissimilar, existing simulators both in-house and at Ft. Rucker, Alabama (1-2). None of these devices had been designed for networked training.

Two key MULTISIM concepts are pertinent to the issues which are the subject of this paper; interoperable simulations and maintaining crew workloads.

Interoperable Simulations

Provisioning to allow the networking of todays and tomorrows training devices demands analysis of the requirements for a total system solution as opposed to merely developing a physical data

link. To provide a comprehensive network configuration, almost all traditional simulation functional areas must be considered including visual/sensor simulations, navigational simulations, tactical simulations, communications, instructional and observational systems and security provisioning. Does a systems analysis dictate that corresponding subsystem simulations be identical in the devices to be networked? In some cases identical simulations or at least identical operational data will be required. However, in most cases interoperable simulations will be adequate.

For example, there will be many adverse inconsistencies in the networked training environment if the databases for visual, sensor and threat simulations are not identical (or nearly identical) in each of the devices on the network (3). However, it may not be necessary for all devices to apply the same fidelity in employing those databases. More specifically scene content management may be applied to selectively display tactically important information which is correlated to other devices because it is taken from an identical database. A tank simulator may need to display only a few kilometers of imagery while a low altitude airborne sensor may require information representing much larger distances. However, the tank must appear on the ground to the airborne sensor and the tank's crew must see the same level of local detail as the airborne sensor observes around the tank. This is necessary to insure that the tank does not appear to drive through trees, buildings, etc. and to insure that the tank will appear properly occulted when partially or fully masked. In this example, the databases must be nearly identical while the simulations operating in conjunction with the databases are interoperable.

Another example of interoperable simulations would be in the area of navigational systems. Common geography modeling may be

essential (4), however, two devices may be interoperable with different Doppler system simulations if each is designed to the accuracy required for the device and each is capable of operating properly with the network geographic coordinates and the localized magnetic variation.

The question, in general, becomes how do we determine where identical or duplicate simulations are required and where interoperable simulations are adequate, and subsequently what are the acceptable fidelity differentials in the interoperable simulations.

Maintaining Crew Workloads

The guideline promoted by MULTISIM is the concept of maintaining the same crew workloads in the simulator during networked operations that the crew would experience when operating the real weapon system in actual combat. In simulator development this translates into providing the realism (or fidelity) required to create and maintain a combat workload. Such workloads can be tremendous in today's advanced weapon systems, especially complex aviation systems which require highly skilled, coordinated operations, split second decision making and which are also inherently unforgiving of mistakes. For example, the individual and coordinated crew workload to employ a Hellfire missile involves rapidly recognizing, identifying, acquiring, tracking and engaging a distant moving vehicle while maneuvering the aircraft at low levels to avoid threats while maintaining sensor line-of-sight to the target being engaged. Oversimplifying the simulations controlling any of the functions involved (i.e., acquisition, tracking, etc.) could allow the crew to obtain simulated performance levels which might not be obtainable under similar real-world conditions or which at a minimum could not be achieved in the same time frames or with the same amount of

weaponry. In a networking environment this negative training could ripple into the perceived team performance negating the effectiveness of the crew's team training and potentially leading commanders to misperceive the capabilities of their forces. Furthermore, if these networked devices are to be used for weapon and avionics systems evaluation the oversimplification can result in inaccurate research data.

Recommendations

It is a goal of those involved in the networking standardization process to provide protocols that will accommodate a large diversity of training systems including a large variation in fidelity levels. It is therefore apparent that users will have to impose limitations on fidelity differentials between devices employed on the network for specific training applications. Moreover, the allowable fidelity differentials will vary dependent on the training objectives desired. If a training session is to be organized to instruct basic team communication skills then communications systems fidelity must adequately modeled in each device while the fidelity relationships of other subsystems may be inconsequential to the intended training. However, if the objective is advanced combined-arms mission training then the total combat workload for each type of device involved must be adequately simulated. A universal tri-services mechanism and an associated governing organization will therefore be required to classify devices relative to functional fidelity and to provide and maintain a database listing which devices can be faithfully operated together for different types (levels) of network supported training. Also, a mechanism and controls will be required to coordinate device upgrades which affect network performance.

The goal of MULTISIM development has been to provide networking capabilities to accommodate the highest fidelity levels available to the aviation community. More specifically if multiple high-fidelity devices are interconnected then their high level of performance will be maintained in networked operations. If lower level devices are connected, their fidelity levels will not necessarily be increased, however, their participation will not reduce the performance levels of the high-fidelity devices. The same network performance is not necessarily required to support crew devices with less demanding workloads (5).

In view of the time, effort and money being expended to create a networking standard, the resulting protocol should ideally be capable of fully accommodating the highest fidelity devices available today and robust enough to expand to the requirements imposed by tomorrows devices. However, judging by the quantity and complexity of issues we face, such a goal may not be achievable in the immediate future. The forthcoming networking standard should therefore allow for existing and future specialized networks by treating them as dissimilar systems and requiring that they be interoperable rather than restricting their potential performance with a specified protocol. More specifically, the standard should allow for networks designed at proprietary and varying fidelity levels to be interoperable via translation systems (or gateways) which must be compatible with the standardized protocol. Such an approach allows interoperability and growth potential yet does not restrict engineering efforts to optimize networking configurations especially at the localized levels. This is an important consideration since existing simulators and a good percentage of future devices will be primarily employed on a day-to-day basis for individual crew and limited team training. The frequency of device networking for large scale exercises will be limited due to the logistics of organizing and executing such activities.

Therefore, based on utilization, localized network performance should have priority. In some instances localized performance may require very high speed, real-time interfaces to meet time critical requirements (6) which might not be obtainable with the standard protocol. Also, on the other end of the spectrum, it may not be cost effective to implement the resources required to support the full protocol when interconnecting a low-level training system of several workstations or part task trainers.

In summary, the costs and controls necessary to allow successful interoperability of DoD simulations can be minimized by judiciously assessing networked operational requirements at a systems level to determine where identical participant simulations are required and where the concept of interoperable simulations can be applied. Once interoperability is achieved, the issue of employing multi-fidelity devices can be addressed by applying the concept of maintaining crew workloads commensurate with the intended training objectives. These concepts apply to the networking of existing simulators as well as to the networking of future multi-vendor devices. These concepts also apply to interconnecting larger scale training systems which may be composed of multiple training devices linked by proprietary or specialized networks.

References

1) George, G.R., Knight, S.N., and Monette, R.; "Multiple Simulator Networking (MULTISIM) - The Way to Provide Effective Combat Training Today"; I/ITSC, Orlando, FL., 1988.

2) George, G.R., Knight, S.N., and McTiernan, J.; "Networking of Full-Fidelity Simulators for Advanced Mission Training"; NAECON, Dayton, OH 1989

3) Schwlam, S.R.; "Correlation of Environmental Databases for Networked Simulators"; Second Workshop for Interoperability of Defense Simulations, Kissimmee, FL 1990.

4) George, G.R.; "Environmental Correlation in Networks of Simulators"; Second Workshop for Interoperability of Defense Simulations, Kissimmee, FL, 1990.

5) George, G.R., Knight, S.N., Stark, E.A.; "Fidelity Requirements for Aviator Training Networks", AIAA Flight Simulation Conference, 1989.

6) George, G.R.; "Time/Mission Critical Issues for Networks of Simulators"; Second Workshop for Interoperability of Defense Simulations, Kissimmee, FL, 1990.

Correlation of Environmental Databases for Networked Simulators

Steven R. Schwalm
Sr. Systems Engineer
CAE Link Flight Simulation

An Issue Paper Prepared
for the Second Workshop
on Standards for Interoperability
of Defense Simulations
Kissimmee, FL 15-17 January 1990

Introduction

The correlation of environmental databases is extremely important to high fidelity simulators. Much work has been done to correlate the visual and sensor databases to provide accurate training for sophisticated aircraft. The networking of devices greatly increases the need for close correlation between databases. The correlation between databases for environmental interrogation or feature correlation such as line-of-sight to targets, crash detection, or laser range finding is also very important.

The issue here is not necessarily the fidelity of the simulation but the believability of the network simulation which greatly affects the training value. This problem exists between devices of various types from different or even the same vendor. The issue of varying the fidelity of devices on the network only complicates the problem.

Major Issues

The following issues must be addressed for the dynamic environment.

1. Height Above Terrain Correlation
2. Crash Detection
3. Line of Sight
4. Visual Sensor, and Automated Threat Databases

Background

Actual experience on the AH-64 CMS (1) and Multiple Simulator Networking (MULTISIM) by CAE Link (2-4) provides the basis for most of the content for this paper. Other networking sources which were referenced are listed (5-7). Outside references on feature correlation are very limited (8).

Height Above Terrain

As players in ground vehicles move across the database, each simulator calculates its own position and attitude and passes it out across the network. If the terrain databases of two simulators from different vendors do not match very closely, then when one of the of the simulators visual model is displayed on the other systems image generator, that visual model could be floating in air or worse yet penetrating the ground. On some image generators, this penetration could cause additional scene generation anomalies which are extremely distracting to the trainee.

Some possible solutions for the terrain following problem are as follows:

1. Pass only the latitude and longitude or similar coordinate data of each simulated player out across the network and make each simulator calculate the altitude and orientation of the displayed visual model. This approach would greatly increase the processing required by each simulator to position and orient each model.

2. Have a high degree of correlation between the databases of all of the simulators on the network. The degree of correlation which is required must yet be determined but it most likely must be less than 1 meter. A one meter error in the altitude of a common passenger car on the ground would have it either penetrating halfway into the ground or floating that same distance in the air.

Crash Detection

Current simulators calculate crashes for the aircraft when that aircraft intersects objects within the environmental database as well as other networked aircraft or players. Several differing methods can be used to calculate these crashes, which I will not go into here, but all of them rely on objects stored within the database. If the databases between networked simulators do not contain the same information to the same level of detail, then anomalies on the network will occur. For example, consider two helicopter simulators. The

second simulator is located behind the first simulator in a tail chase position. They are both flying nap of the earth when the first helicopter, a lower fidelity device which has lower data detail, passes through a high tension pole line. The second helicopter, a higher fidelity device, follows the first helicopter but crashes because he intersected the power lines. This may seem to be a trivial point, but it greatly impacts the believability of the simulation especially for the crew that crashed and thus effects the training value in a negative way.

Line of Sight

Line of sight calculations determine for the host computer the capability of a player to see another player. These calculations are used heavily for automated forces to determine whether the automated force can acquire or shoot at any other player. The correlation of objects within the database can greatly affect believability of the automated forces if the effect of their line of sight requests do not correspond with what a player sees through his visual and sensor systems. Two different problems exist which affect the line of sight calculations; missing objects, and object placement accuracy.

The amount of data in a database greatly affects the fidelity of automated threats. In stand alone simulators, the line of sight for threats is usually calculated by the image generator of the ownship. This provides correlation between threat actions and what the student sees out his windows or sensors The networking of common devices also usually retains this capability. But with the networking of different devices, the commonality between feature correlation systems is lost. The networked devices now use different databases and most likely methods for determining line of sight including the exclusion of various objects.

As an example assume two simulators exist on a common network. If one of the systems, lets say an automated forces unit, does not represent individual trees or buildings within its database, then a helicopter simulator which may be hovering behind a small house out of sight of the automated optical threat, would be seen by the automated force as being positioned out in the open. The helicopter would then become

cannon fodder for the automated threat. This does not only apply to automated forces, but to devices with varying image generation capabilities.

The correlation of the placement of objects within the environmental databases of various simulators can also become critical in the believability of the simulations. For example, consider a tank which is a hundred meters from a building. Another player is positioned three kilometers away so that his line of sight is obscured by the building (Figure 1). Now if the building in the automated tanks database is off by one meter, there is potentially a 30 meter error in the tank line of sight calculation at the 3 kilometer range of the second player (Figure 2). If the range to the target increases, so does the error linearly. This means that the player could be shot by the tank, when in his visual system he can not even see the tank which is an optical threat.

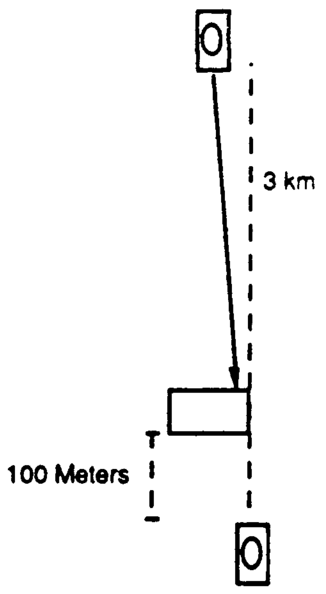


Figure 1

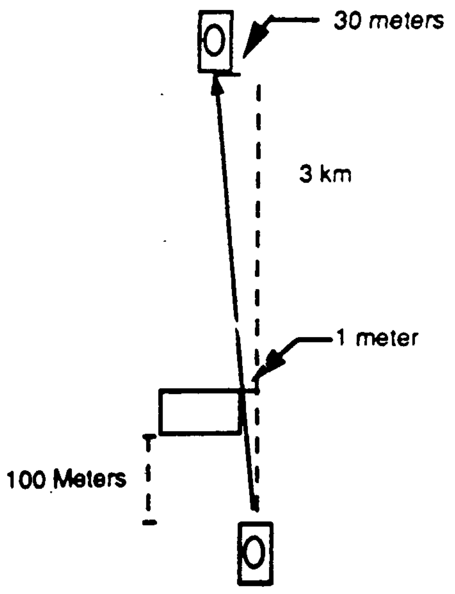


Figure 2

Sensor, Visual and Automated Threat Databases

Networking alters the normal operating practice of a simulation device, which is calculating or displaying only what is important to itself. It now makes each device consider details that impact other devices. For this reason, all of the databases on the network (i.e. visual, sensor, and tactical) should really contain identical information or else anomalies will occur in networked exercises. This information must match in content, but not necessarily in structure. The amount of data used by each device may also differ in both resolution and retrieval area based on the capabilities of the real world device being simulated.

Consider two different simulators, each with different visual systems. Assume the first simulator's player sees himself masked safely behind some trees. The second simulator, since it's visual system can not display trees, observes the first simulator's player without obstruction. This definitely gives the second player an unfair advantage.

Also consider that the databases of the automated threat systems could match identically, but if they are not representative of the displayed visual database, then they have limited training value. The threat databases must represent all of the features that reside in the visual and sensor databases to insure proper threat interaction across the network. The representations may be simplistic, but they must be there. For example, if the threat database did not maintain the individual trees that reside in the visual database, then when an automated player moves across the terrain he will pass through trees that are displayed by the image generator along with the target model. Although this may not be tragic, it would greatly affect the training of someone trying to track the target in his sensor system.

Recommendations

Tactical simulations as required for team training need feature correlation. Furthermore, networks of simulators require correlation between their feature correlation databases. This correlation, since it is used to calculate events which affect the ownship's survivability, become

critical in the believability of the the networked simulation and team training objectives. The correlation requirements must be investigated for all devices, from command consoles to high fidelity helicopters, to determine what tolerances are acceptable from a training stand point.

Two different approaches exist to solve the problems stated above.

1. One network feature correlation device - This would involve placing one, or several of the same type feature correlation devices on the network. These devices would then be responsible for all environmental interrogation necessary for the network. This approach would greatly increase the network traffic, while adding substantial delay to the request responses. It also contradicts the distributed processing philosophy of the SIMNET network.

2. Insure a high degree of correlation in content and placement between all of the feature correlation devices on the network along with the visual and sensor and automated threat databases. This will require a great deal of investigation as to what amount of error is acceptable, but is most likely the most reasonable approach.

Accurate feature correlation is particularly important for low flying aircraft that base their survivability on using the terrain and cultural features to their advantage. For this reason feature correlation between devices on the network must be closely investigated to insure that our databases are accurate enough to support it correctly.

1. Drew, E., George, G.R., and Knight, S. N.; "AH-64 Combat Mission Simulator Tactical System", AIAA Flight Simulation Conference, 1987.

2. George, G.R., Knight, S. N., and Monette, R.; "Multiple Simulator Networking (MULTISIM) - The Way to Provide Effective Combat Training Today", I/ITSC, Orlando, FL., 1988.

3. George, G.R., Knight, S. N., and McTiernan, J.; "Networking of Full-Fidelity Simulators for Advanced Mission Training"; NAECON, Dayton, OH 1989.

4. George, G.R., Knight, S. N., Stark, E. A.; "Fidelity Requirements for Aviator Training Networks", AIAA Flight Simulation Conference, 1989.

5. Demers, W.A.; "Son of SIMNET"; MILITARY FORUM, October 1989.

6. Michelson, Chad; "US Army Simulation Training: No longer A Game of Solitaire"; November, 1989.

7. Schneider, Ronald, "Simulation and Team *Training*"; *Army* Aviation; October 1989.

8. Bondzeit, F., and Edwards, R. E.; "Image Generation for Rotary Wing Applications", I/ITSC 1988.

006-01-90

Dynamic Environment Concerns
for Networked Simulators

Steven R. Schwalm
Sr. Systems Engineer
CAE Link Flight Simulation

An Issue Paper Prepared
for the Second Workshop
on Standards for Interoperability
of Defense Simulations
Kissimmee, FL 15- January 1990

Introduction

In war, the environment does not remain static. It is altered by nature and man from before the first shot till the last casualty has been counted. These environmental changes provide significant cues to the participants as to the current and previous action in an area.

Future network systems in support of team training and mission rehearsal will need to allow for the modification of the environmental database to portray the modifications which have occurred. The occurrence of these events introduces new problems which previously have not been covered by the SIMNET protocols.

Major Issues

The following issues must be addressed for the dynamic environment.

1. Bringing players in late to the war - How do we update a new players database to tell it what has been going on before he joins.

2. Method for identifying environmental objects - The objects within the environment must be tagged to allow their modification during a networked simulation.

3. Environmental effects - Who is responsible for sending PDU's to show the database updates necessary due to environmental changes such as rain and snow, temperature change and variance in barometric pressure.

4. Destructive effects - How do we alter the database in response to explosions and weapon impacts. Also who is responsible for secondary effects like the chain reaction explosions of neighboring fuel storage units or munition dumps.

5. Engineering effects - How do we update the terrain to show the database changes due to engineering units altering a hill or building a bridge.

Background

This paper is based on the extrapolation of training concepts from the CAE Link Multiple Simulator Networking (MULTISIM) program (1-3).

Expansion of Issues

Late players

Participants which enter the war or game after it has begun, need to know the changes which have occurred to the environment since the simulation has begun. The new participant would need to know all of the bridges, buildings, and terrain which have been destroyed or modified since the current networked war simulation began. Consider a tank commander who enters a war on the second day. Two hours earlier an air strike by the opposing force occurred against a strategic bridge that is on the commanders designated route. If the war simulation is to be fair, then when the tank commander reaches that bridge, it had better be destroyed.

There are several basic approaches that could be taken to implement these considerations.

1. Use one common environment database for the network- This method would have one common database on the network, with each network node accessing that database to calculate its visual, sensor, feature correlation, and weather data. This method would require a very high band width network to supply the various nodes with all of the environmental database information.

2. Each system announcing its own modifications - This method requires each simulation to broadcast any change he makes to the database to all of the other systems. This method would require that each simulator record all of the modifications that it has made and retransmit them any time a new player joins the simulation.

3. A database manager node - A specific node on the network would be given the job of managing the database activities. The node would record all of the changes which have occurred to the environment base on the area of the database they are in. When a new player joins, the database manager would broadcast the previous database changes to the new player.

Identifying Environmental Objects

If the system must be able to modify objects, it must then be able to identify them. The objects should most likely be identified by the area of the world in which they reside.

Environmental Effects

Nature modifies the environment constantly. Temperatures change, precipitation occurs, and floods alter the size of rivers. The environment uniquely exists for each player within the network, but elements like visibility must correlate between players (4).

Destructive Effects

Players within the simulation modify the environment with their weapons as well as their existence. The effects of weapons such as the bombing of a runway, can be easily understood. Similarly a tank column running down a dirt road can drastically change the characteristics of the road surface which would impact the maneuverability of those vehicles that follow. These destructive effects must be initiated by each player, but cumulative effects may need to be controlled by some form of a database manager.

Engineering Effects

Along with destruction, man can also create. Engineering units are an integral part of modern warfare. The rebuilding of a bridge or the reconstruction of a bombed runway must be broadcast throughout the network to allow for the units to use it or so that someone can destroy it again.

Recommendations

The implementation of a dynamic environment on a simulator network would greatly enhance the realism of the network in a prolonged simulation. The requirements for the implementation of such an environment must be investigated closely. Several items which must be considered follow.

1. Is a separate environment database manager necessary to keep track of updates, and to make the environment interactive.

2. Will the dynamic terrain add an excess amount of data to the network to the extent of requiring the utilization of an additional environmental network.

3. How will players join the simulation at any time they choose.

The incorporation of a dynamic environment, if implemented correctly, could lift the realism of simulation to a level never before achieved, the type of realism required for effective mission rehearsal and large scale combined arms exercises.

References

1. George, G.R., Knight, S. N., and Monette, R.; "Multiple Simulator Networking (MULTISIM) - The Way to Provide Effective Combat Training Today", I/ITSC, Orlando, FL., 1988.

2. George, G.R., Knight, S. N., and McTiernan, J.; "Networking of Full-Fidelity Simulators for Advanced Mission Training"; NAECON, Dayton, OH 1989.

3. George, G.R., Knight, S. N., Stark, E. A.; "Fidelity Requirements for Aviator Training Networks", AIAA Flight Simulation Conference, 1989.

4. George, G.R.; "Environmental Correlation in Networks of Simulators", Second Workshop on Standards for Interoperability of Defense Simulation, Kissimmee, FL, 1990.

Position Paper: On the definition of Object Types in SIMNET protocol

January 9, 1990

Christina L. Pinon
Institute for Simulation and Training
University of Central Florida
12424 Research Parkway
Suite 300
Orlando, FL 32826

The SIMNET Object Type Numbering Scheme is a very efficient way to define a number of objects using the same group of bits. Presently the Object Type is defined by a 32 bit Unsigned Integer which is interpreted a variety of ways, depending on the type of object being represented. The flexibility of this method is limited, however, by the number of bits. Interpretation may also be aided by arranging similar fields along the same bits.

My proposal is to increase the number of bits in the "ObjectType" field from 32 to 64 bits. There are several issues that I see related to a change of this type. These are:

1) The effect of the increase in bits in relation to the existing PDU's.

2) The provision for expansion of classes and subclasses as the number and types of objects being simulated increases.

1. The effect of the increase in bits in relation to the existing PDU's

There are several considerations to address when attempting to change the bit definitions in SIMNET Protocol. One is the need to follow guidelines for bit alignment requirements. SIMNET protocol has provided restrictions on the size of certain data elements in order to meet the bit alignment requirements of certain computers. Another consideration is the size of the PDU. Although traffic on the network is currently not a problem, it is an issue to be examined for future, large scale simulations.

In the former case it is evident that any change would affect the form of a number of PDU's. Adjustments would have to be made in terms of padding or rearrangement of fields. This has been done recently when the protocol was adjusted to allow for multiples of 64 bits in the simulation and data collection PDU's. It therefore seems feasible that an adjustment for a larger object type is possible without a significant change from current

protocol requirements. The choice of 32 additional bits is to make adjustments to the present form of the PDU's easier since some PDU's have 32 bit padding. Others can be made to conform by adding 32 bits of padding.

In the latter case, increasing the size of a PDU could affect traffic on the network. Studies would have to be done to show just what kind of effect it would have. It has been shown in the past (by studies done by BBN..see Report No. 6711) that network traffic is relatively low. It appears that increasing the object type to 64 bits would have little effect on the overall network traffic.

2. The provision for expansion of classes and subclasses as the number and types of objects being simulated increases.

The object type in SIMNET protocol is used to define different types of objects in the simulated world such as vehicles, projectiles, ammunition, bridges, etc. Vehicles and Munitions are two object types that require more detail than other objects and probably have the most variety in class, subclass, model and function. Presently, some of these fields are nearly used up. By adding another 32 bits to the number there would be more than enough room to include new fields and additional bits dedicated to current fields. This would allow almost endless flexibility for future development in object types.

Justification for this change can be given in the following example:

Consider the Munitions field of the Resupply Variant.

The munitions field is an object type. The following is a description of what is represented by the different bits in this 32 bit number and some observations concerning them.

As it currently exists, only 3 bits are allotted for "Domain" (a limit of 8 choices). 5 choices are already used. Most objects fall in the existing categories, however, other categories may be created in the future to either more specifically describe an object that is presently defined as a subset of an existing domain or for data collection purposes where defining a specific domain may be important for analysis purposes. This may remain 3 bits if domain is redefined in a more general way and the next few fields (class, subclass, etc) are increased to allow more specific definition.

The next 4 bits represent "class". This gives a total of 16 choices (17 if 0 is included). Currently 5 (6) are used. Depending on how specific one wishes to be on classes (one can be less specific and make use of the subclass category) this may or may not be adequately large enough. I believe that there may be

enough new classes in the future to warrant making this field bigger.

The next 5 bits represent either "caliber" or "target" depending on the value of the "class" field. If it is "caliber", the existing definition appears to be adequate. If it is "target", the choices that exist seem to better represent a "subclass" rather than a target. The number of bits for this field, however, are sufficient as they now stand.

"Subclass" is described by the next 4 bits. The definition for this field is determined by the value in the "class" bits. 4 bits only allows 16 (or 17) choices to be represented. Presently for a class of 4 (projectile) there are already 12 values defined. Expansion of this group of bits is recommended.

The "country" field is sufficiently large. Perhaps more choices could be defined than are presently represented.

The "model" and "series" categories are represented with the last 10 bits. These, I believe, are sufficiently large.

See attached figures for suggested changes.

Conclusion

In its current form, the definition of Object types in the SIMNET protocol allows for a large variety of objects within the same 32 bit field. There is not, however, much room for expansion of these bit fields.

The increase in the number of bits in the "object type" field is one way that the SIMNET protocol could be made more flexible for future growth in network simulation. It also opens the doors for the use of the same protocol for non-military applications which may require more specific definition of objects to be simulated.

The choice of adding 32 bits is to aid in maintaining the bit alignment requirements established by the SIMNET protocol. In some PDU's the adjustment will be as simple as eliminating added padding or adding an additional 32 bits of padding to "round-out" the PDU.

One other suggestion for the object type is to line up similar bit definitions. If two objects contain a description for model, have those group of bits be in the same part of the 64 bit field.

Recommended Format for the Object Type Numbering Scheme for SIMNET Protocoal Data Units

VEHICLE

DOMAIN	CLASS	SUBCLASS	ENVIRONMENT	COUNTRY	SERIES	MODEL	FUNCTION
BITS 1-8	BITS 9-16	BITS 17-24	BITS 25-32	BITS 33-40	BITS 41-48	BITS 49-56	BITS 57-64

AMMUNITION

DOMAIN	CLASS	SUBCLASS	CALIBER	COUNTRY	SERIES	MODEL	FUNCTION
BITS 1-8	BITS 9-16	BITS 17-24	BITS 25-32	BITS 33-40	BITS 41-48	BITS 49-56	BITS 57-64

MISSILE

DOMAIN	CLASS	SUBCLASS (TARGET)	WARHEAD	COUNTRY	SERIES	MODEL	FUNCTION
BITS 1-8	BITS 9-16	BITS 17-24	BITS 25-32	BITS 33-40	BITS 41-48	BITS 49-56	BITS 57-64

OTHER

DOMAIN	CLASS	SUBCLASS	THESE FIELDS TO BE DETERMINED ACCORDING TO THE OBJECT BEING REPRESENTED				FUNCTION
BITS 1-8	BITS 9-16	BITS 17-24	BITS 25-32	BITS 33-40	BITS 41-48	BITS 49-56	BITS 57-64

** NOTE THAT THE ABOVE FIELDS HAVE ALL BEEN EXPANDED TO 8 BITS IN ORDER TO FORM UNIFORM BIT BOUNDARIES AT 8 BITS EACH.

Protocol Data Elements and Protocol Data Units affected by a change in the Object Type

1/9/90

by Christina L. Pinon

Basic Data Elements

- Burst Descriptor
  - projectile
  - detonator
- Munition Quantity
  - munition
- Vehicle Guises
  - distinguished
  - other
- Vehicle Status
  - vehicleType
  - specific
    - VehicleSpecificStatus
      - category
        - SpecificStatusCategory
          - genericVehicleStatus
            - Munition Quantity
              - munition

Simulation Protocol Data Units

- Activate Request
  - guises
    - Vehicle Guises
      - distinguished
      - other
  - status
    - Vehicle Status (see vehicle status above)
      - vehicle Type
      - munition
- Vehicle Appearance
  - guises
    - Vehicle Guises
      - distinguished
      - other
- Fire
  - burst
    - Burst Descriptor

- <u>projectile</u>
- <u>detonator</u>

- specific
  - fire type shell
    - <u>ammoSelected</u>

- Impact
  - burst
    - Burst Descriptor
      - <u>projectile</u>
      - <u>detonator</u>

- Indirect Fire
  - burst
    - Burst Descriptor
      - <u>projectile</u>
      - <u>detonator</u>

- Resupply
  - vehicleType
  - munitions
    - Munition Quantity
      - <u>munition</u>

<u>**Data Collection Protocol Data Units**</u>

- Vehicle Status
  - status
    - VehicleStatus (see vehicle status above)
      - <u>Vehicle Type</u>
      - <u>munition</u>

** Underlined elements are the specific object types.

New Burst Descriptor Sequence
(based on Object Type as a 64 bit Unsigned Integer)

```
type BurstDescriptor sequence {
     projectile          ObjectType,
     detonator           ObjectType,
     quantity            UnsignedInteger (16),
     rate                UnsignedInteger (16),
                         unused (32)
}
```

PROJECTILE	DETONATOR	QUANTITY	RATE	UNUSED
Object Type - 64 bit Unsigned Int.	Object Type - 64 bit Unsigned Int.	16 bit Unsigned Integer	16 bit Unsigned Integer	32 bits

New Munition Quantity Sequence
(based on Object Type as a 64 bit Unsigned Integer)

```
type MuntionQuantity sequence {
     munition            ObjectType,
     quantity            Float (32),
                         unused (32)
}
```

MUNITION	QUANTITY	UNUSED
Object Type - 64 bit Unsigned Int.	32 bit Float	32 bits

New Vehicle Guises Sequence
(based on Object Type as a 64 bit Unsigned Integer)

```
type VehicleGuises sequence <
    distinguished        ObjectType,
    other                ObjectType
>
```

DISTINGUISHED Object Type - 64 bit Unsigned Int.	OTHER Object Type - 64 bit Unsigned Int.

New Vehicle Status Sequence
(based on Object Type as a 64 bit Unsigned Integer)

```
type VehicleStatus sequence <
    vehicleType          ObjectType,
    odometer             Float (32),
    age                  UnsignedInteger (8),
                         unused (24),
    failures             VehicleSubsystems,
    specific             VehicleSpecificStatus
>
```

VEHICLE TYPE Object Type - 64 bit Uns. Int	ODOMETER 32 bit Float	AGE 8 bit Unsigned Int.	UNUSED 24 bits	FAILURES Multiple of 32 bit sequences	SPECIFIC Multiple of 32 bit sequences

ABSOLUTE TIME STAMP IN NETWORKING OF SIMULATORS

Amnon Katz, Ph.D.*
McDonnell Douglas Helicopter Company
Mesa, Arizona

ABSTRACT

The communications delays inherent in networking of simulators, especially over long haul links, present a problem in close interaction of remotely simulated players. The problem is most obvious in formation flying. Attempting to hold position wingtip to wingtip, the delay tends to cause each player to believe the other is lagging behind. At fighter speeds the effect is considerable. The solution is to correct for the delay by extrapolation. This requires an absolute time stamp for dynamic variables that are sent. The receiving simulator performs an extrapolation from the time stamp to current time. The source, form, and required accuracy of the time stamp are discussed.

I INTRODUCTION

The transmission delay can play a significant role in the networking of simulators. This is particularly true for long haul networking. A delay corresponding to the speed of light is a hard minimum imposed by the laws of nature. Speed of light delay is 3.33 usec per kilometer, which amounts to 5.33 msecs per thousand miles. The delay can easily be doubled when the actual rates of propagation in communications lines are used. Further delays are caused by switching and other equipment. In case of a satellite link, the round trip to geostationary altitude imposes a minimum delay of 200 msec, and the mechanics of the equipment on the satellite increases this to half a second or more.

An aircraft travelling at 400 knots covers one meter in about 5 msecs. Position discrepancies due to communications delays are visible in close formation flying even for good links over several thousand miles. Satellite links would make either close formation or close combat impractical.

The obvious remedy is to compensate by extrapolating the received data over the period of the delay. But delays over communications lines are not always predictable and repeated precisely. It is necessary to extrapolate each packet of data over the delay it experienced. The mechanism to achieve this is to include a time stamp with the variables of state of each data packet. The stamp is the time for which the variables are valid as opposed to the time at which they were computed or transmitted. In the terminology of Katz et al in reference 1, the time stamp is "Dynamic Time". The receiving node subtracts this time from the time at which the variables are to be displayed and extrapolates over the difference.

* Manager, Advanced Development Office, Combat Simulation and Sys. Int.

Obviously the mere act of extrapolating cannot compensate for very long delays. The permissible delays will be bounded by the predictive power of the extrapolation algorithm. It stands to reason that no algorithm can reliably predict the future actions of a human pilot. Thus the delays permissible in close formation or close combat will be limited to the human reaction time of a fraction of a second. Without time stamping and extrapolation, however, close formation and close combat between widely separated simulators is ruled out altogether.

II THE TIME STAMP

For the time stamp to achieve its purpose, it is necessary for all networked players to possess synchronized clocks. This should be achieved by each player independently sysnchronizing his clock to his local time zone or to Universal Time Coordinated (UTC). This should allow players to join and leave the networked game without requiring a specific time check.

It is suggested that the time stamp be the time elapsed since the beginning of the current hour. This convention serves to eliminate differences in clock settings related to time zones and seasonal clock shifts. Remotely located networked simulators can be situated in different time zones. The locality may or may not implement daylight saving time. The facility may choose to synchronize with local time or UTC. None of these factors influence the time stamp.

Simulation display update rates vary and sometimes are asynchronous. Still 60Hz is an accepted standard for high fidelity simulations. This amounts to a 16.7msec frame. Delays of several frames in the internal mechanization of simulators are not uncommon. Still, for the time stamp to achieve its purpose it must be smaller than a frame and serve to determine the frame to which data belong. I suggest an accuracy of 1msec as both sufficient and achievable.

The simplest method to represent the time stamp is as an integer. My suggestion for the scaling is to set 2^32 equal to one hour (3600sec), which makes each unit represent 3600sec/2^32 = 0.838usec. This scaling is convenient for the following reasons:

1. The time stamp fits in one long word.

2. The time stamp naturally rolls over at the end of each hour.

3. The resolution is better than the suggested one millisecond accuracy by a factor of about one thousand, leaving room for future improvements.

III SOURCES

The National Bureau of Standards (NBS) maintains an accurate time standard and distributes it by radio broadcasts from its stations WWV, WWVB, and WWVH. The broadcast signal (reference 2) can easily be decoded with an accuracy of 1msec. The signal is itself subject to propagation delay at the speed of light. This can easily be corrected, using the known constant distance from the radio station to the simulation facility. Commercial off the shelf equipment exists for decoding the radio signal and interfacing it with digital

computers.

Telephone services that provide time signals for digital computers are also available. The National Institute of Standards and Technology (NIST) offers a telephone service that can correct for propagation delay over the telephone lines (ref 3). This requires that the receiving node echo each byte received. When this is done at 300 baud, an accuracy of 1msec can be maintained. The telephone service has a time limit for each connection. The service would have to be accessed repeatedly as necessary.

In designing the synchronization scheme of a simulation node, both the accuracy and the stability of the on board timing device must be considered. A tolerance of 1msec maintained over 1 hour amounts to 10^-3/3600 = 2.8*10^-7 or better than 0.3 ppm. This exceeds the accuracy of most crystal oscillators, that is the tolerance by which their actual frequency deviates from their nominal frequency. However, so long as the stability of the oscillator is better than 0.3 ppm, the actual frequency can be determined and used in computing the time stamp, and the time count need be reset only once an hour.

The above discussion shows that a time stamp accuracy of 1 msec is achievable with state of the art equipment. The implementation details may be left to each facility.

IV SAFEGUARDS

An erroneous time stamp can cause more harm than good. Before a time stamp is used to determine the interval over which variables of states are to be extrapolated, certain checks should be applied. These are based on comparing the time stamp (TS) to the time at which the packet is received (TR) and to the time at which the information is to be used (TU):

1. $$A < TR-TS < B$$

The interval TR-TS roughly amounts to the propagation delay. If this exceeds a reasonable limit B (of the order of 1 second), the stamp must be assumed wrong. In this case one may do better by substituting some constant estimate of the delay, i. e. replacing TS by TR - DEL.

- 3 -

A lower limit A must also be applied to the delay. If the time stamp represents the transmission time, then the delay cannot be less than the propagation time or in any case cannot be less than zero. However, it is permissible for the sending node to compute for a time slightly in the future. In this case the receiving node needs extrapolate over less than the propagation delay, or even interpolate slightly backward in time. But again, if TR-TS is less than some value A (say of the order of -0.5 sec) the stamp must be assumed wrong and a standard delay applied.

2. $$TU-TS < C$$

TU-TS is the interval over which extrapolation is to be applied. If this interval exceeds a limit C (in the range of 2 to 5 seconds), the data packet is obsolete and it must be assumed that the connection was lost. In that case

the bound C should replace the interval. This would have the effect of freezing the remote player at a position he would reach a time C after the time stamp of the last packet received. The fact that the connection has been lost is then advertized by the player stopping dead in its tracks or in mid air.

No lower limit need be placed on TU-TS. TU is of necessity later than TR, and therefore the limit A in condition 1 automatically applies.

3. TU-TS < D

If the last packet is older than D (of the order of minutes up to half an hour) the remote player must be assumed to no longer be in the game. In this case action should be initiated to remove it.

The conditions, corrective actions, and limits are summarized in the tables below:

Table 1: CONDITIONS

No.	Condition	Corrective action if violated
1.	A< TR-TS < B	TS:= TR - DEL
2.	TU-TS < C	TS:= TU - C
3.	TU-TS < D	Eliminate player

Table 2: CONSTANTS

A	Lower bound on propagation delay	-0.5sec
B	Upper bound on propagation delay	1 sec
C	Upper bound on extrapolation interval	2-5 sec
D	Max time before player who lost data communications is eliminated	5-30 minutes

V SUMMARY

An absolute time stamp for data packets in networked simulation is both practical and useful. Synchronization of each node can be achieved independently by use of time signals disseminated by NBS and NIST. Inexpensive off the shelf equipment for assimilating these signals is available. The absolute time stamp makes it possible to compensate correctly for packet propagation delays. Coupled with the concept of Dynamic Time, the time stamp makes possible an accurate accounting of all time discrerpancies within the sending and receiving node as well as the propagation medium. An optimum compensation for time delay thus becomes possible. Applications where networked players interact closely, such as formation flying, are impossible without compensating for time delays.

REFRENCES

1. A. Katz, D. Allen, and J. Dickson, "Synchronization and Time Tagging in a Distributed Real Time Simulation", AIAA Simulation Technologies Conference,

Boston, August 1989 p 259.

2. S. L. Howe, "NBS Time and Frequency Dissemination Service", NBS Special Publication #432, September 1979.

3. Information available from the NIST Office of Standard Reference Material, B311 - Chemistry Building, NIST, Gaithersburg, MD 20899, 301-975-6776.

END OF DOCUMENT

22 December 1989

MEMORANDUM FOR: Dr. Dexter Fletcher, Suite 707 (SIMNET Facility),
1911 N. Fort Meyer Drive, Rosslyn, VA

SUBJECT: SIMNET Database Issues

1. Per the request of Mr. George Lukes, ETL, the following brief comments pertaining to SIMNET data usage are provided. The discussion is very brief and many more issues exist. However, the issues we raise seem important and worthy of your consideration.

2. Level of Detail (LOD): The LOD can dramatically affect the usage of SIMNET. If mission rehearsal is a program goal, then the LOD must be very high and artificial LOD may not be entirely appropriate. As an example, if repeated training sessions lead soldiers to depend on the presence of key terrain features, their real-world performance could depend on those same features. A tank crewman might find a good location for a hull defilade position in the SIMNET world and depend on that location in the real world. (Note that the absence of that concealed position may not be readily apparent in the real world.) This can be a genuine problem. The current state of available digitized data is not at a sufficient resolution to accurately depict such positions when concealment comes from slight depressions on the terrain. The LOD of objects in SIMNET is also important; an example could be based on the target recognition task performed by air defense gunners.

3. Interfaces: The LOD concerns interfacing SIMNET to the trainee. The converse issue, interfacing the trainee to SIMNET is also very important. If a standard map is presented to the trainee, then his/her conception of the SIMNET world will not mesh entirely with the terrain in the SIMNET world. SIMNET data, a spin-off of Defense Mapping Agency Interim Terrain Data (ITD), does not depict all information available on a standard 1:50,000 map sheet. The Tactical Terrain Analysis Data Base (TTADB), from which ITD are developed, may not be updated as frequently as topographic maps; TTADB information is often dated, particularly the transportation overlay which is critical to SIMNET. Moreover, the ITD will be created from merging data from different TTADB sources which were not created based on a registration standard appropriate for computer usage. To experience either of these observations, take a TTADB and lay it down on the map it describes. The data will not register and some data will not appear on both source materials. This is acceptable when a human uses the map, but a computer will have to sort out these issues. This is not a straight-forward task, computationally. A present solution is to supply the user with a map generated from the SIMNET data (instead of a topographic map); this map is more in line with the presented world-view. Different "overlays" (corresponding to the TTADBs) could be provided and a shaded relief map could be generated from SIMNET data. The bottom line revolves around the use we intend to get from SIMNET. As a trainer, map differences are inconsequential. As a mission planner, differences among the real world, the topo sheet, and SIMNET could be critical. The same problems might be encountered in preparing for such events as tank gunnery competition.

4. Raster-based or Polygon-based: The data must be flexible. At times, it will be most appropriate to view the data as raster information (for pixel-based path planners as an example). At other times, a polygonal (or areal) representation would be far better, both for computational speed and realism (a fly thru the terrain as an example). Conversion from areal data to point data is well defined and reasonably fast; the converse is not true, so dynamic data conversion could be quite difficult. Further, the point at which raster data should become areal data is very fuzzy. Should elevation points be grouped into (areal)

mountains or into collections of peaks which can be grouped to form mountains. Should mountains be viewed as individuals or as part of mountain ranges. Further, at what point on the ground does the mountain end and the valley start? How far down the hill is the military crest? These could be real considerations for the SAFOR, depending on the type of functions they should be able to complete autonomously. In fact, SAFOR functionality needs to be defined before data representation is decided. Data availability needs to be ascertained before SAFOR functionality can be decided.

5. Multiple Simulator Models: Different simulators (JESS, JANUS, etc) do different things with the raw data provided to them. If they are to be allowed entry into the net, some very serious consequences could result (not only based on data but on interpretation of the data as well). As an example, mutual line of sight, achieved at the same time, has to be ensured between different simulators. A way to address this issue might be to devise a set of situations that different simulator models must "pass" before they can play on the net. This would be similar to Ada compiler validation.

6. Data Manipulation: Many "views" of the SIMNET world will be possible and useful. Aside from data manipulation to compute line of sight, coverage, etc, data manipulation can be used to model the effects of external forces on the terrain. Weather, seasons, engineer actions, the use of obscurants are all important issues which affect the user view of the world. These issues, if addressed in SIMNET, will also have to be come into play for other simulators (on the net). In essence, there are two sorts of data manipulation; one doesn't change the data itself and one does. Both are important. The latter case will cause additional problems with distributing the world view to nodes on the simulation network.

7. In summary, several key issues are:

- ITD comes from low resolution, inaccurate, and dated TTADB sources. This data can not exactly describe the real world.
- SIMNET needs multiple views of the same data (points and objects). The data must be flexible, hierarchical and pre-stored.
- The user needs a clean, high-fidelity interface which standard paper maps can not provide.
- Data availability will drive SAFOR functionality which will drive data representation.
- Different simulators behave differently. A validation test is needed.
- Different types of data manipulation need to be addressed. Functions that make changes based on the data need to be standard and distributed. Functions that change the data need to be centralized and the results distributed.

8. The point of contact for further information is MAJ Robert Richbourg who may be reached at AV 688-4871 or (914) 938-4871.

GERALD E. GALLOWAY, Jr.
Colonel, Professor, USMA
Head, Department of Geography

Position Paper

Proposed Changes to the Vehicle Appearance PDU

JORGE CADIZ
Research Associate

The Institute for Simulation and Training
12424 Research Parkway, Orlando, FL 32826

January 10, 1990

University of Central Florida

Position Paper

Proposed Changes to the Vehicle Appearance PDU

Introduction

This position paper refers to specific changes that are being recommended for the Vehicle Appearance Protocol Data Unit (VA PDU).

This PDU is used to describe a vehicle's changes in appearance through time. A large percentage of the traffic in a SIMNET distributed simulation consists of VA PDUs.

There are two changes proposed in this paper which will allow for greater flexibility, and provide room for future expansion. Specifically, the proposed changes are:

- The addition of 32 bits to the Vehicle Appearance field
- The elimination of the Stationary field, and placing of a Stationary flag in the Vehicle Appearance field

Rationale for the Proposed Changes

Vehicle Appearance Field:
In order to support a large scale simulation with many different types of simulators, we must be able to model these simulators with a certain amount of detail. To achieve this task certain features of the vehicles which are to be modeled must be defined. The SIMNET protocol, as it exists, allows for features to be modeled in two ways:

1. Using the *Vehicle Specific* field.
2. Placing flags in the Vehicle Appearance field.

The *Vehicle Specific* field in the VA PDU takes on various forms that are dependent on the *Vehicle Class*. Currently, there are three vehicle classes defined; Static, Simple, and Tank. These classes are determined by the number of independently moving parts that are modeled in a vehicle. If we are

to describe additional vehicles using the *Vehicle Specific* field, then new classes must be created.

Another method that may be used to define new vehicle attributes, is to use the Vehicle Appearance field. Currently one third of this field is being used to model the appearance of certain ground vehicles. This field should be increased from 32 bits to 64 bits to allow for future expansion. Items such as air vehicle attributes should be taken into account in this field. These attributes will be defined as new simulators are made interoperable, and the need for greater appearance fidelity becomes an issue. For example, there is the need to allow for objects such as landing gear, flaps, in-flight refueling probes, and other moving parts of air vehicles to be modeled in a simulated exercise. The specific positions of these components can be expressed in the Vehicle Appearance field similar to the technique used to describe the position of the M2 TOW launcher in the current SIMNET Protocol [1].

Stationary Field:
This field should be eliminated. If the vehicle's velocity vector is zero, it will be expressed as a flag in the Vehicle Appearance field. The stationary flag is to be used as a tool in smoothing algorithms that will provide for fluid movement of vehicles in the simulation. Eliminating this field and placing a Stationary flag in the Vehicle Appearance field will allow for a savings of 16 bits.

Conclusion

The VA PDU makes up a large portion of the traffic during a SIMNET simulation exercise. Increasing the size of the VA PDU by several bytes should have a nominal effect on network bandwidth. We must allow for as much flexibility and room for future expansion as possible. The expansion of the Vehicle Appearance field proposed in this paper should allow for some of this flexibility. At the same time we must proceed cautiously, and try to minimize the traffic on the network. This can be done by condensing the protocol, and making efficient use of the fields available.

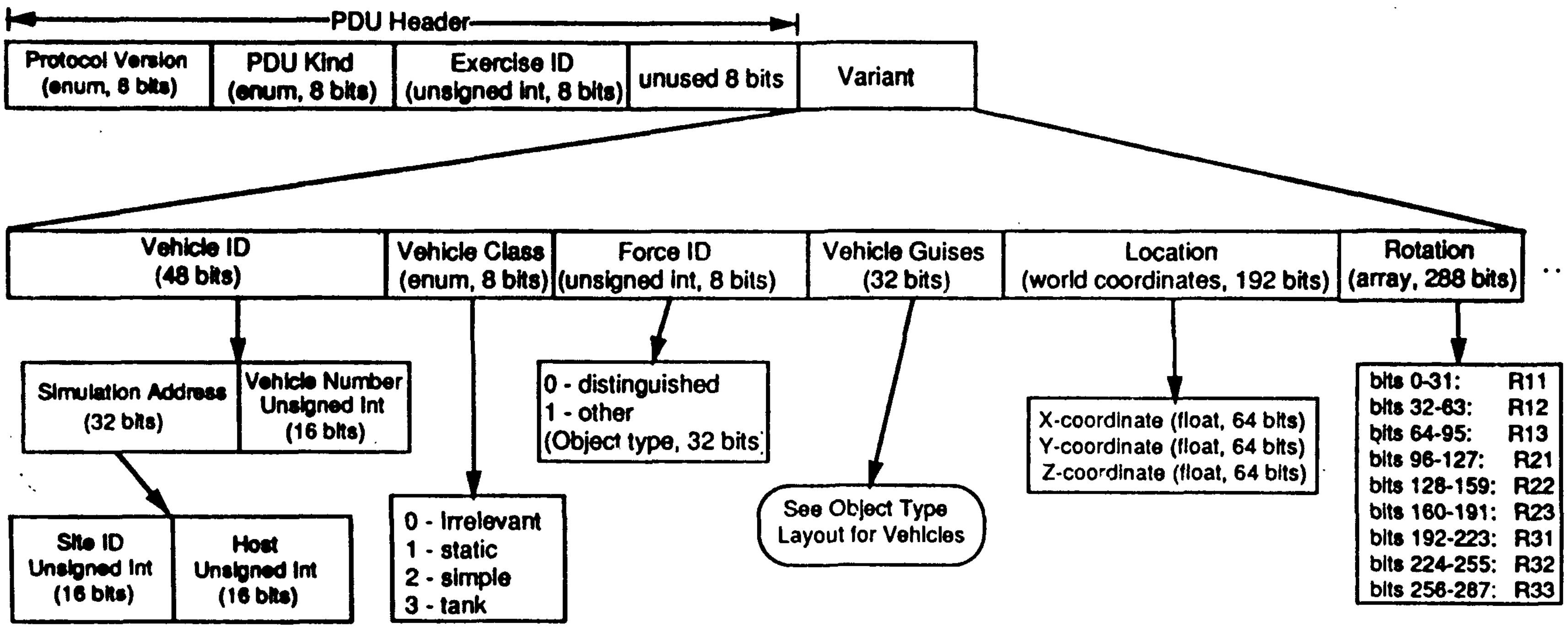
Attached is a diagram of the suggested VA PDU format. Also included in the attached figure are changes to the SIMNET Protocol which will affect this PDU in SIMNET Protocol version 6.0 [2].

BIBLIOGRAPHY

[1] A. Pope, "**The SIMNET Network and Protocols**", BBN Technical Report No. 7102, July 1989.

[2] C. Kanarick and A. Pope, "**Summary of SIMNET Protocol Changes**", BBN Technical Report (Draft), January 1990.

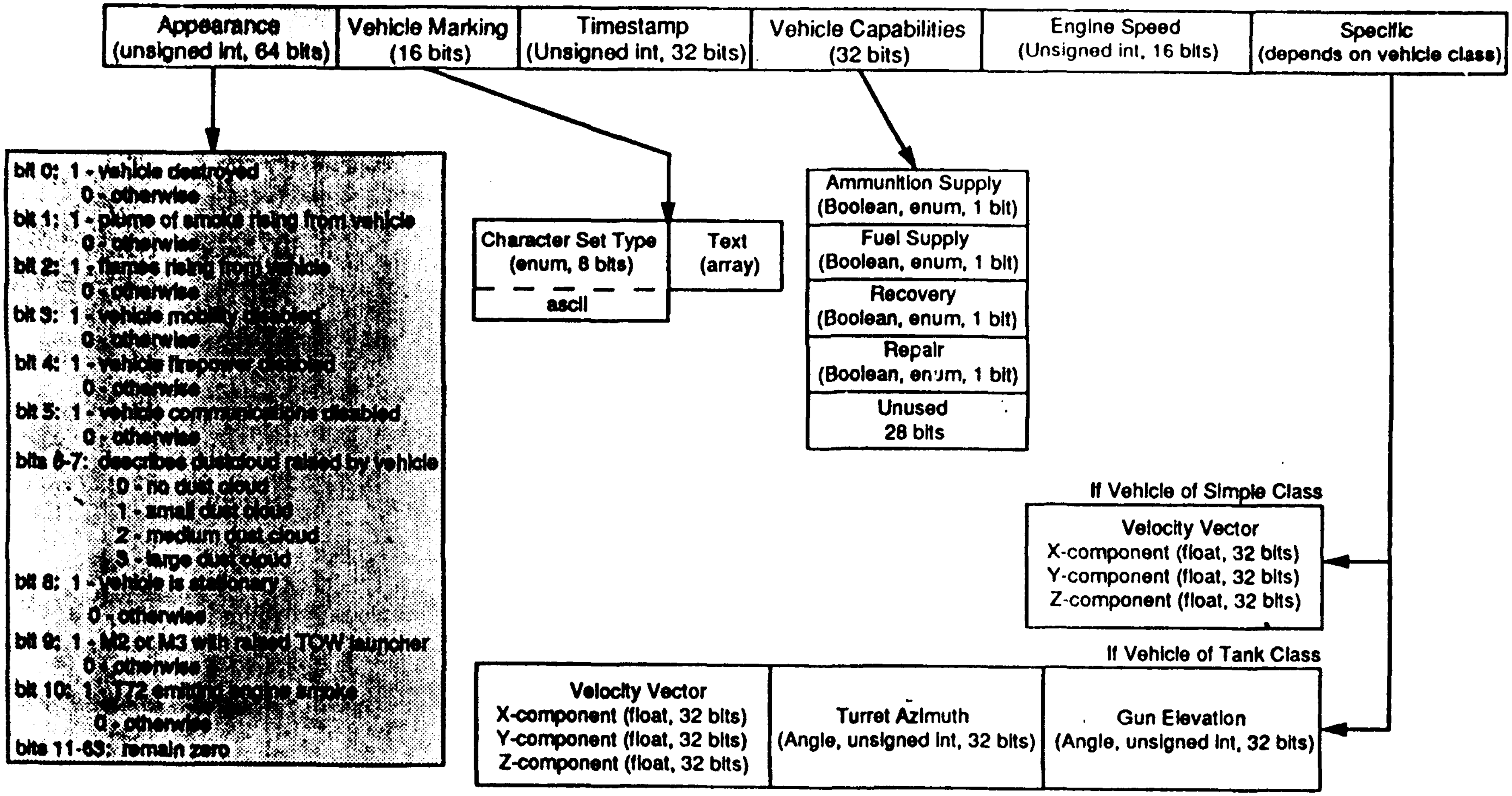
Figure 1. Vehicle Appearance Protocol Data Unit



Fields affected by suggested changes

Jorge Cadiz - 12/20/89
IST/UCF

Vehicle Appearance Protocol Data Unit (con't)



Fields affected by suggested changes

Jorge Cadiz - 1/10/909
IST/UCF

World Coordinate System

Robert Glasgow
Institute for Simulation and Training
University of Central Florida

A means of reporting position in three-space is necessary for interactions between simulated vehicles operating on, above, or below the surface of the earth. The current SIMNET protocol expresses position as a three-element array of 64-bit floating point numbers [1]. A flat gaming area is assumed, and the origin of the coordinate system is an arbitrary point such as the southwest corner of a 30 km by 30 km terrain database. The first array element might represent displacement in the east direction, the second north, and the third up. Such a coordinate system, while adequate for land-based vehicles, is clearly unsuitable for aircraft or surface vessels whose visual range is affected by the curvature of the earth. Submersible vessels which might operate below the reference datum must be considered as well. A world coordinate system is needed which can effectively represent vehicle positions in three-space.

There are several properties of our planet which suggest trade-offs between the selection of any world coordinate systems. Using a traditional latitude-longitude-altitude coordinate system necessarily involves a great deal of transcendental function calculation, and introduces the issue of what datum would serve as the altitude reference. Given its oblate shape, a line perpendicular to a tangent plane ("altitude") is not necessarily parallel to the effect of Earth's gravity ("down"). For aircraft in particular, this involves additional small angle calculations in the computation of vehicle forces and moments. For high fidelity, real-time simulation systems, the floating point operations required to perform these calculations must be considered a scarce resource.

This paper is intended to illustrate the tradeoffs involved with two methods of reporting position in world coordinates. One conserves network traffic at the expense of additional floating point calculations, and vice versa.

There are further issues of precision, data word alignment, and conversion between coordinate systems which must be considered in a thorough analysis of this topic.

Position Along a Subtended Arc

The radius of the earth at the equator is given to be 6,378.077 km [2]. At 50 km (164,042 ft) altitude, a circle surrounding the globe with circumference of $2\pi(6,428,077 \text{ m}) = 40,388,799$ m may be imagined. Such a circle is introduced in order to represent the maximum length dimension which could possibly require measurement in the simulated world. The IEEE standard for the 64-bit floating point data type yields 15 digits of precision, which implies position reporting to ±50 nanometers along this circle. This is clearly more precision than conceivably necessary for any type of vehicle training device.

The IEEE standard for the 32-bit floating point data type yields 7 digits of precision. This yields ±5 m of position reporting capability which is not sufficient.

A technique to halve the network traffic imposed by each position update is to use 32-bit integers to represent latitude and longitude. The 32-bit integer represents the fractional part of a circle, bisected successively 32 times until a resolution of $2^{-32} = 1/2^{32} = 2.3283\text{E-}10$ is obtained.[1] Applying this fraction to the circumference of the earth at 50 km altitude, a precision of (40,388,799 m)(2.3283E-10) = 9.40 mm = 0.370 in. (This technique is used in the SIMNET protocol to express the turret angle relative to the front of the hull.)

Altitude can then be represented by a 32-bit signed integer representing height above (or below) a datum in millimeters. The IEEE standard for a four-byte signed integer yields a range of ±1,334 miles with a resolution of 1 mm.

Orthogonal Cartesian Space

Position can be represented in an orthogonal Cartesian space, with the origin at an arbitrary point (e.g., center of the earth). A right-handed coordinate system can be established with the x-axis intersecting the equator at the prime meridian, the y-axis intersecting the equator at 90° E, and the z-axis intersecting the north pole. Vehicle positions are thus reported in components of x, y, and z which can be readily manipulated with vector arithmetic. Range calculations would be simplified from using transcendental functions to the Pythagorean theorem.

The components of x, y, and z could be cast as 64-bit floating point, which is compatible with the present SIMNET format.

---

[1] Such an approach is not uncommon in the simulation industry. Although not yet appearing in the standard mathematical literature, this technique is loosely referred to as the binary angle measurement (bam).

Computational Issues Related To World Coordinate System

Consider the computation inherent with each position update: (1) A moving model calculates an updated position, (2) the position is translated from local to standard coordinates, if different, (3) the coordinates are broadcast over the net and received by other nodes, (4) the position is translated into the local coordinate system, if different, and range calculations are made, (5) sensor input (radar, visual, etc.) is generated if the new position is determined to be observable.

Clearly, a considerable amount of computational resources are devoted to position determination and range calculations. The world coordinate system selected must not impose an excessive burden upon the computing power of the simulation host.

The introduction of dead reckoning models into the SIMNET protocol has significantly reduced the number of position updates which must be broadcast. Experience suggests that network traffic is no longer a critical factor, and that emphasis should be placed upon protocols which lend themselves to efficient computational strategies.

Conclusion

A great many alternatives for a standardized world coordinate system may be considered, each with its relative merits and disadvantages. For a low-cost, uniprocessor-based distributed simulation system, the processing contingent upon the selection of any one protocol must weigh heavily in the evaluation.

A standardized world coordinate system will have a significant effect upon every simulation adhering to that standard. The decision as to which will be adopted must be based upon a carefully researched analysis, and represent the best combination of relative attributes.

References

[1] Kanarick and Pope, "Summary of SIMNET Protocol Changes," BBN Systems and Technologies Corporation, Cambridge, MA, Jan 1990.

[2] Chemical Rubber Company, *CRC Handbook of Chemistry and Physics*, 60th ed (Boca Raton, FL: CRC Press, 1979).

U12-...

Position Paper: On Adopting the SIMNET Local Area Network Deactivate Request and Response PDUs in the Local Area Network Standard

January 10, 1990

Karen Danisas
University of Central Florida
Institute for Simulation and Training
Suite 300
Orlando, Florida 32826

The SIMNET protocol as it now stands allows for the withdrawal of any vehicle from the simulation. This is performed by issuing Deactivate Request and Deactivate Response PDUs. These PDUs include the headers plus the particular variants.

The Deactivate Request variant consists of the following series of fields:

```
type DeactivateRequestVariant sequence
     {
     Vehicle ID
     Reason
     8 unused bits
     }
```

The Deactivate Response variant is issued in response to a Deactivate Response PDU. This variant consists of the following fields:

```
type DeactivateResponseVariant sequence
     {
     Vehicle ID
     result
     8 unused bits
     }
```

a. Vehicle ID - 48 bit sequence.

In both variants there is a Vehicle ID field. Each vehicle in the exercise has a unique vehicle ID. This field consists of two parts: a 32 bit simulation address that identifies the simulator modeling the vehicle, and a 16 bit unsigned integer vehicle number that uniquely distinguishes that vehicle from all others present in the simulators. BBN has incorporated this new representation in their latest version and it allows for increasing the number of vehicles without increasing the number of bits.

b. Reason - 8 bit enumeration.

This field describes reasons for the deactivation request and response. The eight bits seem to be adequate for the possible number of reasons.

c. 8 Unused Bits

These bits will remain unused to keep the proper structure of the PDU.

It is recommended that these two PDUs be adopted in to the Local Area Network Standard.

Position Paper: On Adopting the SIMNET Local Area Network Activate Request PDU in the Local Area Network Standard

January 10, 1990

Karen Danisas
University of Central Florida
Institute for Simulation and Training
12424 Research Parkway
Suite 300
Orlando, Florida 32826

Activate Request PDU Variant

The placement of any vehicle in to the simulation is performed by issuing an Activate Request PDU. This PDU includes the header plus the particular Activate Request PDU Variant.

There are several different data elements that are included in this PDU that currently do not allow for increasing the number of vehicles to play on the network. What follows is a list of the fields in this PDU and where recommended changes should be made.

```
ActivateRequestVariant sequence
(
Activation Reason
Vehicle class
Vehicle ID
Organizational Unit
Vehicle Marking
Vehicle Guises
Simulated Time
Terrain Database ID
Vehicle Status
On surface
31 unused bits
Location - World Coordinates
Specific - 64 bits
)
```

a. Activate Reason - enumeration type of 8 bits.

This field gives the reason for the simulator to be activated. The number of bits seem to be adequate.

```
type ActivateReason enum (8)
(
activateReasonOther
exerciseStart
exerciseRestart
vehicleReconstitution
towingArrival )
```

b. Vehicle Class - enumeration of 8 bits.

Each vehicle is classified according to how many independently moveable parts it has, and what algorithm should be used to dead reckon it's appearance. This field should be increased to 16 bits in order to accommodate an increase in the number of different classes of vehicles that may eventually play on the network.

```
type VehicleClass enum (8)
{
vehicleClassIrrevelant
vehicleClassStatic
vehicleClassSimple
vehicleClassTank
}
```

c. Vehicle ID - 48 bit sequence.

Reference Position Paper on Deactivate Request and REsponse PDUs.

d. Organizational Unit - 32 bit sequence.

This field tells how each vehicle is associated with a series of organizational units and the hierarchy of command. The first element in the field is the force ID. This is an 8 bit unsigned integer that tells what force that vehicle belongs to. It usually is two forces, however, the setup allows for 256 different forces in an exercise. That seems more than adequate.

The next element describes the organization type. This is of an 8 bit enumeration type. This field should be increased to 16 bits for future implementations. For example, we have the Army, Marines, Air Force, etc.

The next element describes the hierarchy of the organizations. This uses an 8 bit unsigned integer field that identifies each unit, and an 8 bit enumeration field to describe the type. This all seems to be adequate.

```
type OrganizationalUnit sequence

    {
    force - ForceID:
         8 bit unsigned integer(0-255)
         special: forceIrrelevant (0)
                  distinguishedForceID (1)

    organizationType - OrganizationType:

    hierarchy - array of (organizationalLevels) of
                UnitIdentifiers
    }
```

where:

```
     type OrganizationType enum (8)
     type UnitIdentifier sequence
     type UnitType enum (8)
     Constant OrganizationalLevels (9)
```

e. Vehicle Marking - 16 bits sequence.

This field consists of an enumeration field of 8 bits that defines the character set, and an 8 bit array of 11 character strings. The character set defines how the text in the strings should be interpreted and displayed. At present only the ascii character set is defined. Eight bits seem to be too many for this element, and 8 bits not enough for the array. It is recommended that one 16 bit array define the vehicle marking and the first four of the bits define the character set type.

```
type VehicleMarking sequence
{
text - array (MaxVehicleMarkingLength) of 16 bit
       unsigned integers
}
```

f. Vehicle Guises - 64 bit sequence.

This field consists of two object type elements. The object type is dependent on the force that that vehicle is assigned to. One is distinguished and the second is labeled "other". The object type sequence itself has been addressed in a Position Paper by Christina L. Pinon, dated January 9, 1990.

```
type VehicleGuises sequence
{
distinguished - ObjectType (32 bit unsigned integer)
other - ObjectType (32 bit unsigned integer)
}
```

g. Simulated Time - 32 bit unsigned integer.

This field is adequate as is.

h. Terrain Database ID - 24 bit sequence.

This field consists of an 8 bit array of 14 characters for terrain database names, and a 16 bit unsigned integer for the terrain version. It's likely that this is strictly SIMNET protocol and does not belong in networking protocols. However, because there is a need for a standard for passing terrain database information, it is suggested that this PDU still contain information concerning the terrain database being used for the gaming area.

```
type TerrainDatabaseID sequence
     (
     terrainName
     terrainVersion
     )
```

This issue cannot be resolved entirely at this time because of the terrain issue inconsistencies. However, it is recommended that these twenty-four bits remain as is to allow some kind of naming scheme to be implemented at a later date.

i. Vehicle Status - 430 bit sequence.

This field consists of the object type element, a 32 bit float for the odometer, 32 bits for the vehicle age, 174 bits to describe the vehicle subsystems, and 160 bits for describing the vehicle specifications. Many of the fields in this sequence have unused bits. Within the subsystems sequence, many fields are boolean and simply describe whether a certain feature is on or off. The extra bits that already exist can easily accommodate additional feature definition.

j. On Surface - 1 bit boolean.

This bit describes whether the vehicle is to be placed on the surface or not at activation time. It seems natural that this bit should just be a part of the previous field. Then the total extra 32 bits can be utilized by the specific field.

k. 31 Bits Unused.

These bits are necessary for keeping the structure of the PDU as presently defined. It is recommended that they be incorporated into the specific field for the standard.

l. World Coordinates - 192 bit float

Reference Position Paper by Robert Glasgow.

m. Specific - 64 bit sequence (Add on multiples of 64 bits)

Depending on the vehicle class, there are either 2 angle types, or one angle and 32 bits of padding. An important issue comes up here that if the vehicle has more than 2 moving parts, there is not enough bits provided to describe the orientation of the additional moving parts. One solution would be to add additional padding at the end of this PDU, knowing that they would be used at a later date.

Position Paper: On Adopting the SIMNET Local Area Network Status Change, Status Query and Status Response PDUs in the Local Area Network Standard

January 10, 1990

Karen Danisas
University of Central Florida
Institute for Simulation and Training
12424 Research Parkway
Suite 300
Orlando, Florida 32826

Status Change PDU Variant

At the time that the operational status of a vehicle or any of its subsystems changes, the vehicle's simulator issues a Status Change PDU describing what has changed and why. The following describes the different fields and the recommended changes.

```
type StatusChangeVariant sequence
     {
     vehicleID - VehicleID
     8 unused bits
     effect - StatusChangeEffect
     cause - choice (effect) of:
          effectVehicleDestroyed:
               kind - DamageCause, + 24 unused bits
          effectVehicleReincarnated:
               kind - RepairCause + 24 unused bits
          effectSubsystemsDamaged:
               kind - DamageCause + 24 unused bits
          effectSubsystemsRepaired:
               kind - RepairCause + 24 unused bits
     Event ID
     Agent ID
     Subsystems
     }
```

a. Vehicle ID - 48 bit sequence

Reference Position Paper by Karen Danisas on Deactivate Request and Response PDUs.

b. 8 Unused Bits

These 8 bits will remain unused to keep proper structure.

c. Effect - 8 bit enumeration

This field describes how the specified vehicle was affected. Four specific effects are currently defined: vehicle destroyed, vehicle reincarnated, subsystem damaged, and subsystem repaired. The 256 choices allowed for are adequate.

d. Cause - 32 bit sequence

This field describes what caused the change of operational status reported by the PDU. This field is 32 bits long, where the first eight describe the kind of cause and the last 24 are unused and there to keep structure. Currently there are four "kinds of causes" defined: destroyed, reincarnated, damaged and repaired; all are 8 bit enumeration types.

e. Event ID - 16 bit integer

This 16 bit number is generated by the vehicle's simulator and is associated with the particular event that that vehicle is involved with. Sixteen bits are more than adequate to describe the events. It is recommended that this field remain as is.

f. Agent ID - 48 bit sequence

The Agent ID is merely a Vehicle ID field. Each vehicle in the exercise has a unique vehicle ID. This field consists of two parts: a 32 bit simulation address that identifies the simulator modeling the vehicle, and a 16 bit unsigned integer vehicle number that distinguishes that vehicle from all others present in that simulator. This representation allows for an increased number of vehicles without increasing the number of bits.

g. Vehicle Subsystems

This field consists of 174 bits that describe the different vehicle subsystems. The way it is set up allows for more choices to be defined depending on the vehicle. Also, within this field the different systems are described using boolean types. There are unused bits there that will allow future attributes to be added. It is recommended that this field remain as is.

Status Query PDU Variant

The Status Query PDU allows the querying simulator to specify from who information should come from and the type of information that it wants. The following describes the different fields and the recommended changes.

```
type StatusQueryVariant sequence
     (
     Response Kind
     Unit Relation
     Simulator Type
     Vehicle ID
     16 Bits unused
     Organizational Unit
```

a. Response Kind - 8 bit enumeration

This field indicates the type of information sought by specifying a Data Collection PDU. The choices are exercise status, vehicle status or simulation status.

b. Unit Relation - 8 bit enumeration

This field allows the query to select respondents based on the organizational units they simulate. It specifies one of four cases: irrelevant, specified, included, or including. The 256 choices available should be adequate.

c. Simulator Type - 16 bit enumeration

Each type of simulation system participating in the simulation is described by a simulator type code. The number of different type that are allowed for is adequate.

d. Vehicle ID - 48 bit sequence

Each vehicle in the exercise has a unique vehicle ID. This field consists of two parts: a 32 bit simulation address that identifies the simulator modeling the vehicle, and a 16 bit unsigned integer vehicle number that distinguishes that vehicle from all others being modeled by that simulator. This representation allows for an increased number of vehicles without increasing the number of bits.

e. 16 Bits Unused

These bits will remain as is to keep proper structure.

f. Organizational Unit - 32 bit sequence

This field tells how each vehicle is associated with a series of organizational units and the hierarchy of command. The first element in the field is the force ID. This is an 8 bit unsigned integer that tells what force that vehicle belongs to. It usually is two forces, however, the setup allows for 256 different forces in an exercise. That seems more than adequate.

The next element describes the organization type. This is of an 8 bit enumeration type. It is recommended that this field be increased to 16 bits.

The next element describes the hierarchy of the organizations. This uses an 8 bit unsigned integer field that identifies each unit, and an 8 bit enumeration field to describe the type. This all seems adequate.

Status Response PDU Variant

When the conditions specified by a Status Request PDU are not met, then the simulator returns a Status Response PDU. It is recommended that this field remain as is.

```
StatusResponseVariant sequence

    {
    result
    56 unused bits
    }
```

Position Paper: On Adopting the SIMNET Local Area Network Activate Response PDU in the Local Area Network Standard

January 10, 1990

Karen Danisas
University of Central Florida
Institute for Simulation and Training
12424 Research Parkway
Suite 300
Orlando, Florida 32826

Activate Response PDU Variant

Any simulator that correctly receives an activate request PDU must immediately respond by returning an activate response PDU. The following fields are included in addition to the PDU header.

```
type ActivateResponseVariant sequence
{
Vehicle ID
result
unused 8 bits
timeLimit
48 unused bits
}
```

a. Vehicle ID - 48 bit sequence

Reference Position Paper by Karen Danisas on Deactivate Request and Response PDUs.

b. Activate Result - enumeration 8 bits

There are currently five results defined indicating whether the request was accepted or not. These eight bits, which allow for 256 reasons to be defined, are more than adequate.

c. 8 Unused Bits

It is recommended that these bits be used to create a new field, 8 bit enumeration, that tells if the activation was successful.

d. Timelimit - 16 bit unsigned integer

The timelimit tells how long the responding simulator will take before it can issue appearance PDUs for the newly activated vehicle. No changes need to be made in this field.

e. 48 Unused Bits

These bits will remain unused to keep the proper structure, and possible new fields.

016-01-70

PROTOCOL PROFILES FOR SIMNET EVOLUTION

AL KERECMAN
USACECOM
JOINT COORDINATION OFFICE
FORT MONMOUTH, NEW JERSEY

PROTOCOL PROFILES FOR SIMNET EVOLUTION

ARCHITECTURE

The SIMNET architecture is based upon Local Area Network (LAN) service between the Simulator/Trainer workstations (S/Ts). Figure 1 shows a possible interconnection of LANs using LAN bridge concepts. This is a simple way of providing broadcast service worldwide over leased lines, and is implementable today with off-the-shelf hardware and software. The bridges set up the interconnections such that redundant paths are not operable concurently; this is probably the method previously employed for interconnection between SIMNET LANs.

BACKGROUND INFORMATION

If each S/T transmits 4 packets of information per second, each packet 256 bytes in length, including error recovery; and if every S/T receives all packets from other connected S/Ts, then each S/T must be capable of processing (receive side) through it's ethernet connection, (n-1)(1024 x 8) bits per second. If each LAN can have up to 80 S/Ts, then any given LAN can be distributing up to (80)(1024 x 8) bits per second, and each S/T receives (79)(1024 x8) bits per second for processing. While the transmit side for each S/T remains somewhat constant, it's receive side loading depends directly upon the number of S/Ts participating in the scenario.

For order of magnitude calculations, using T-1 service (1.544mbps), the total number of supportable S/Ts is: 1.544 x 10E6 / (n)(1024 x 8) or, n = 1544000/8192 = 188. With network administration, diagnostics, and management the supportable number is probably closer to 170. Certainly, multiple T-1 or higher bandwidth lines can be employed for WAN interconnects of the LANs through bridges or gateways.

The limitation of the ethernet LAN is of some significance; running at 1 x 10E7 bits per second, it degrades as a function of loading and when tuned, is able to support perhaps as much as half it's clock rate in traffic loading.

The terminals connected to the LAN are also limiting factors; for example, an IBM/AT machine running at 8 MHz. allows a throughput of 200 KBPS at a packet size of 256 bytes (DEC Networks and Communications Buyers Guide, pg. A-46, July-December 1989). Assuming throughput can be scaled with processor clock speed, a 33MHz. AT machine should be capable of 800KBPS throughput at the same packet size. Therefore, the first order throughput limitation, dependent upon the vintage of the S/T, is probably that of the terminal interconnect to the LAN; and the second order limitation, is that of the LAN itself or the LAN/WAN bridge.

PROTOCOL PROFILES

Looking at the protocol profiles, the present SIMNET is considered a proprietary application layer protocol with integrated functions spanning the five layers below to join with ethernet at the physical/electrical layer (Figure 2). If SIMNET is to run via a connectionless profile, then it should interface via ISO 8206, Connectionless Transport Protocol, at the transport layer. A typical profile would use ISO 8602 at transport, 8473 at network, 8802-2 type 1 at link, and either 8802-3 or FDDI at link and physical layers. If a connection oriented profile is to be employed, a typical profile would embrace ISO 8073 at transport, 8208 at network, 8802-2 type 2 at link, and 8802-3 or FDDI at link and physical layers. Evolution to ISDN services at the physical/electrical and link layers can be accomplished through an interface to ISO 8208 at the network layer. Fax and teletype can be added to the ISDN capabilities via a CCITT I.451/q.931 network interconnect from the LAP D link layer.

EVOLUTION

An approach to evolving from these constraints (figure 3) might be to attack the LANs and the terminal interconnection in an integrated fashion by implementing FDDI. FDDI will function at 100 x 10E6 bits per second with bandwidth allocations for video, data, and digitized voice. Cardsets for the terminals interconnection will possibly provide data at a rate such that the S/T will again be the limiting factor on throughput.

The employment of FDDI should allow time for the SIMNET protocol to evolve into an OSI conforming profile. This strategy can then allow migration into ISDN and beyond. Since the network providing the interconnections of the LANs is assumed to be classified, there is a requirement to evolve the KG from the present 84 capabilities to higher bandwidths, and/or advancing the BLACKER models/techniques. Products using the Secure Data Network System (SDNS) Security Protocol 3 (SP3), now under development, will make provision for a layer 3 security to be implemented in the connectionless mode. Under an SP4 umbrella, connection oriented service security can evolve either from the 84 concept or via integrated security modules inserted into the S/T.

An evolution from the present SIMNET to an OSI based implementation might consist of the following:

1) Establishing a CM process for the SIMNET evolution.
2) Structuring SIMNET into the ISO-OSI seven layer structure (ISO 7498 and CCITT X.200).
3) Establishing OSI network management and administration.
4) Migrating from ethernet to the ISO 8802-3 LAN.
5) Evolving the security functions for both connectionless and connection oriented service.
6) Transitioning from 8802-3 to FDDI using the same link, network, and transport profile.
7) Implementing connection oriented service to interface into ISDN.
8) Employing FTS 2000 services for the interconnection of the SIMNET LANs.

CONCLUSION

Regardless of the evolutionary process steps taken, working closely with NIST and ITS, and embracing and implementing the IEEE, ANSI, ISO, and CCITT standards is the only supportable approach for SIMNET evolution.

HIERARCHICAL RELATIONSHIPS AND INTERACTIONS OF INTEREST

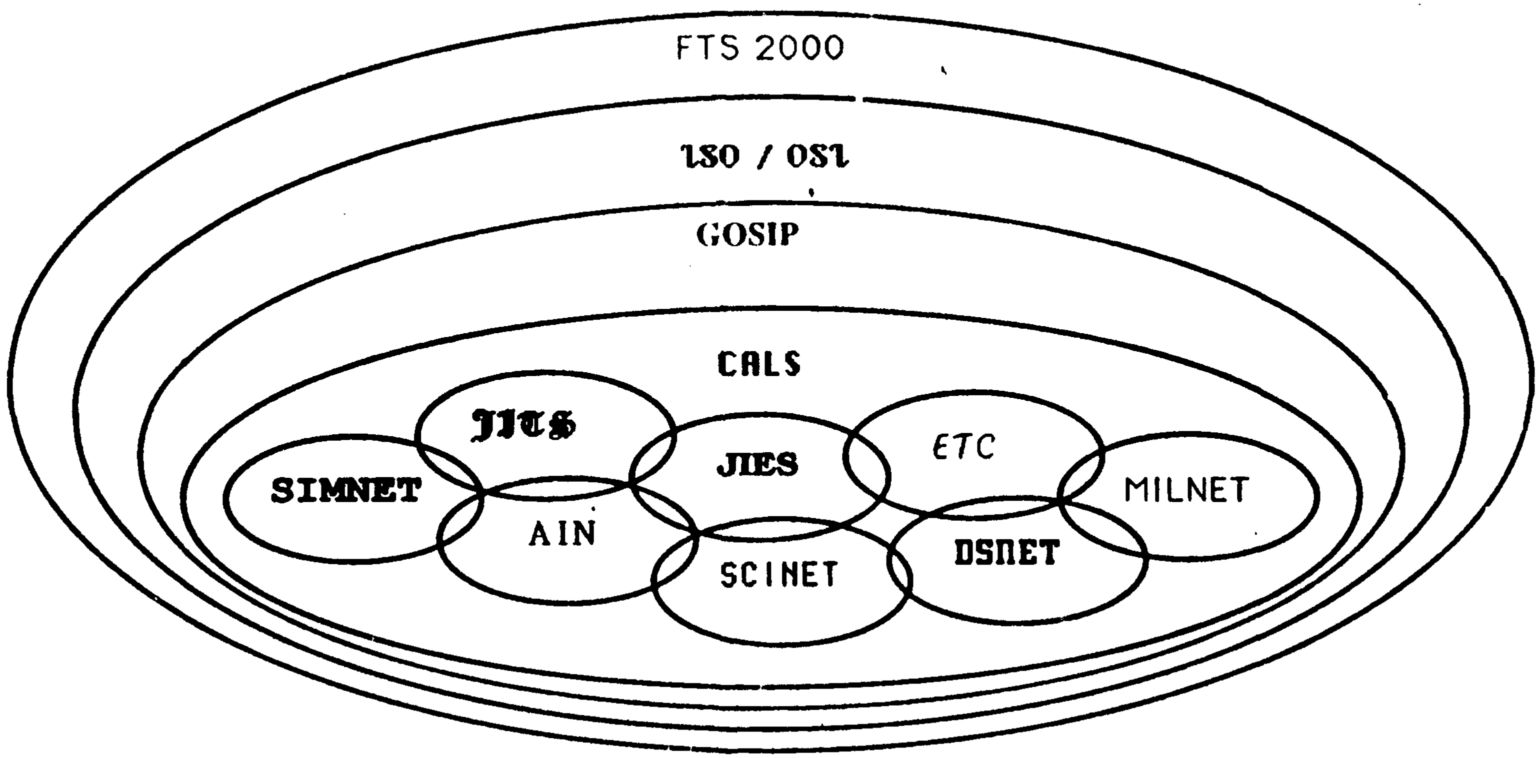


FIGURE 0

ITEMS OF COMMON INTEREST

* CONFIGURATION MANAGEMENT AND CONTROL
* CONFORMANCE AND INTEROPERABILITY TESTING
* APPLICATION SOFTWARE
* DATABASE MANAGEMENT
* NETWORK ADMINISTRATION
* NETWORK MANAGEMENT
* NETWORK CONTROL
* SECURITY
* SERVICES

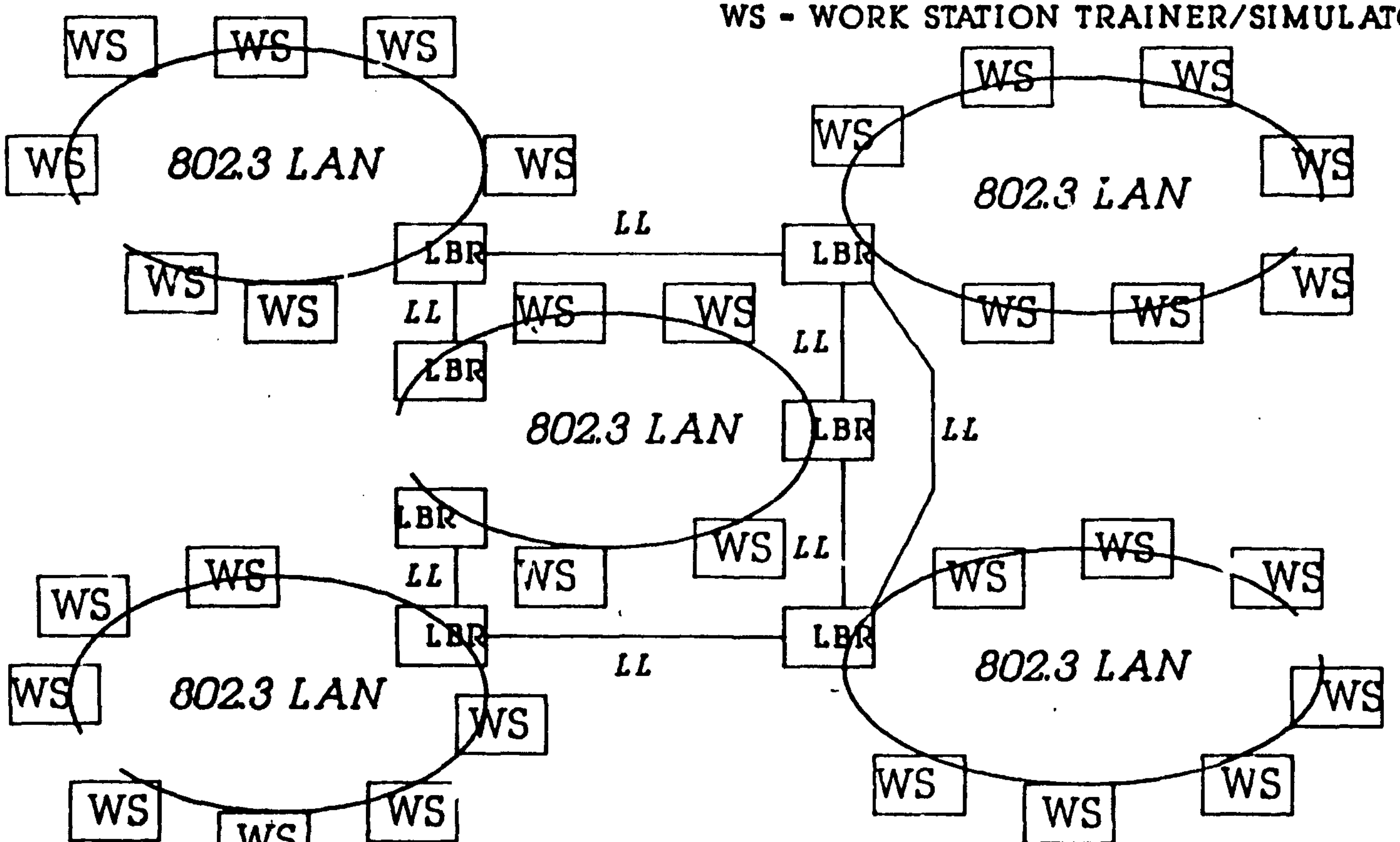
FIGURE 0.1

SIMNET ARCHITECTURE

LL = *LEASED LINES*

LBR = LAN BRIDGE AND ROUTER

WS = WORK STATION TRAINER/SIMULATOR



(256 bytes/packet; 4 packets/sec; (80 WS/LAN) = ((8192 + overhead) bps/WS; broadcast)

FIGURE 1

PROTOCOL PROFILE EVOLUTION POSSIBILITIES

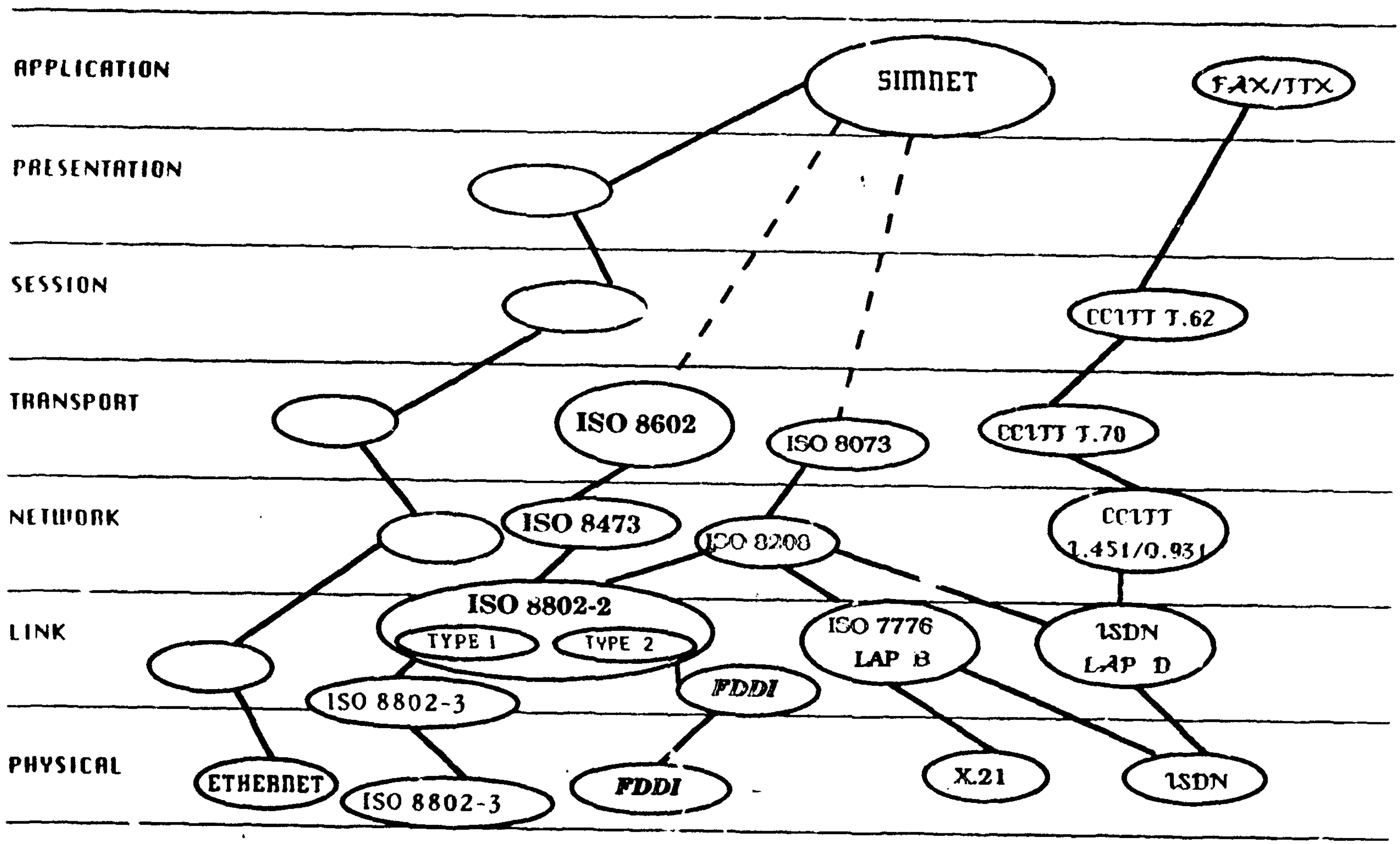


FIGURE 2

SIMNET COMMUNICATIONS ARCHITECTURE EVOLUTION POSSIBILITIES

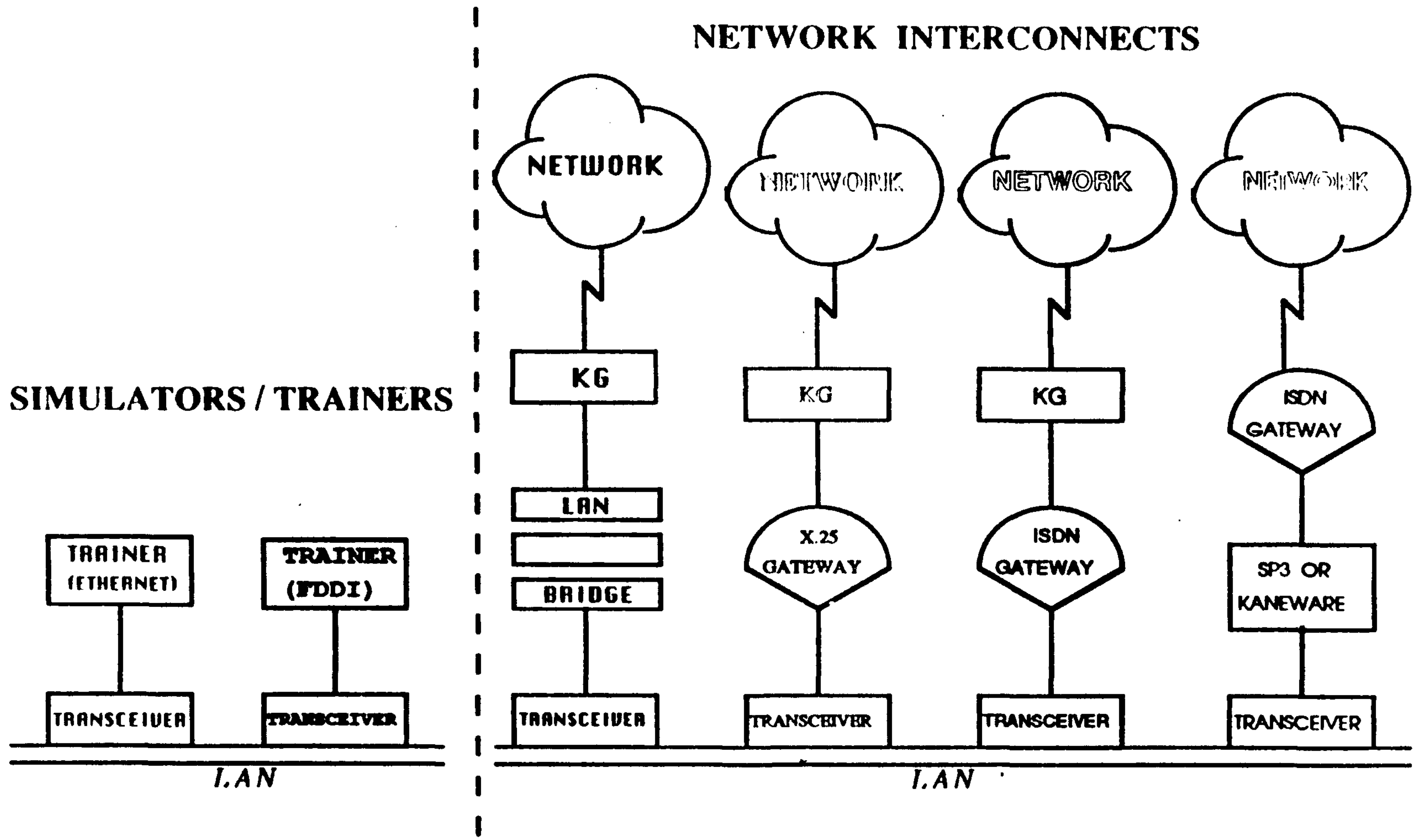


FIGURE 3

(.1 ')- ('(- 9C)

USE OF GLOBAL COORDINATES IN THE SIMNET PROTOCOL

WHITE PAPER ASD-90-10

January 1990

Prepared by:

Jerry Burchfiel and Stephen Smyth
BBN Systems and Technologies Corporation
Advanced Simulation Division
10 Moulton Street
Cambridge, Massachusetts 02138

Submitted to:

NSIA / DARPA / University of Central Florida
Second Workshop on Standards for Interoperability of Defense Simulations
Orlando, Florida
15-17 January 1990

Executive Summary

Previous SIMNET exercises were constrained to take place within a selectable exercise area 75 km on a side. Future SIMNET exercises will take place over much larger areas of the world. For example, exercises planned for 1991 will cover a major part of Europe and include participation by Army, Air Force and Navy forces. In order to support these much larger exercises, we must be able to communicate positions within a worldwide database across the SIMNET network.

However, use of worldwide coordinates in the SIMNET protocol must not significantly increase the computation load imposed on each simulator. The processing required to interpret vehicle appearance update packets received at each simulator is now the critical bottleneck limiting future growth in the size of exercises. This concern is particularly serious when we consider future exercises that may consist of 10,000 to 30,000 vehicles.

We recommend adoption of a Cartesian geocentric coordinate system (Earth centered and Earth fixed) to represent positions, velocities, and orientations within the SIMNET network protocol.

Using the approaches described below, we achieve positional accuracies of a fraction of a meter worldwide, without incurring a significant increase in the runtime arithmetic operations required within a simulator.

1.0 PROBLEM STATEMENT

1.1 Current SIMNET Protocol

Current SIMNET network protocol (Ref. 1) uses appearance packets to communicate the position of a moving vehicle or a battlefield effect (e.g., shell detonation) across the SIMNET network as Cartesian coordinates, x, y, and z, each of which is a 64-bit IEEE Standard 754 floating point number. This representation is adequate to provide a sub-micron position resolution within a worldwide range.

This Cartesian representation also supports efficient simulator implementations, because these coordinates can be used directly within a simulator to drive the computer image generator (CIG) to create images of the other vehicles and battlefield effects.

The further choice of Cartesian coordinates aligned with the local surface of the Earth (topocentric coordinates) provides the additional benefit of being able to choose basis vectors East, North, and Up which are constant vectors over the portion of the battlefield within interaction range. This coordinate system is shown in Figure 1. These vectors are needed in the dynamic vehicle simulation, where the acceleration of gravity acts in the negative Up direction for the platform and its ballistic projectiles. They are also needed in soldier-machine interfaces that display vehicle heading and attitude.

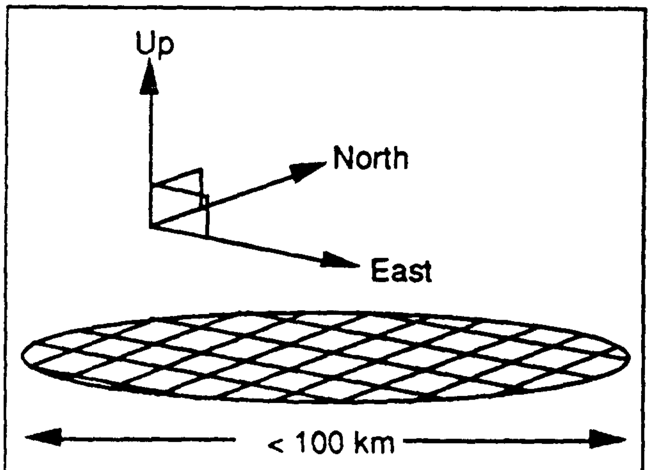


Figure 1. Current SIMNET Coordinate System

1.2 Requirement for a Worldwide Coordinate System

Previous SIMNET exercises were constrained to take place within a selectable exercise area 75 km on a side. Future SIMNET exercises will take place over much larger areas of the world. For example, exercises planned for 1991 will cover a major part of Europe and include participation by Army, Air Force, and Navy forces. In order to support these much larger exercises, we must be able to communicate positions within a world database across the SIMNET network.

This is not only a requirement to support different simulators operating at large geographic distances from each other, but is also a requirement to permit a single simulated vehicle to move continuously across a large part of the world (e.g., fighter bombers flying from England to Libya).

We will be required to support worldwide exercises, which will require use of worldwide coordinates in the SIMNET protocols.

1.3 Requirement for Computational Efficiency

Large exercises planned for 1990 will involve about 1000 vehicles in a single SIMNET exercise. This will result in network traffic estimated at 1000 packets per second, with a size of about 1 kilobit each. If no external "relevancy filtering" is provided to reduce this load, each simulator must receive this packet traffic, and ignore vehicle-appearance packets from vehicles that are beyond the simulator's maximum battlefield interaction range (e.g., visual, sensor, or radar range). Typically, only 100 vehicles will be in view at one time, so this range filtering may reject 90 percent of the traffic received over the SIMNET network.

This filtering must be very efficient so it does not consume too much of the available computation resources of each simulator. The range rejection condition currently employed is:

Reject if:

$$x < x_0 - R \text{ or } x > x_0 + R \text{ or } y < y_0 - R \text{ or } y > y_0 + R$$

where x, y is the location of the remote vehicle, x_0, y_0 is the location of the vehicle being simulated by this simulator, and R is the maximum battlefield interaction range for this simulated platform (typically 5 to 10 km.) This test requires four arithmetic operations (comparisons with precomputed boundaries) per packet, or about 4,000 arithmetic operations per second. The test does not include differences in altitude ($z-z_0$) because current applications only include ground operations, close air support, and nap of the Earth flying, where altitude differences are negligible with respect to R.

Use of worldwide coordinates in the SIMNET protocol must not greatly increase the computation load, particularly if this computation competes for scarce processing resources within a simulator.

For the packets accepted (vehicles within battlefield interaction range), we update the Remote Vehicle Approximation (RVA), which performs a dead reckoning or constant velocity extrapolation of the position of a remote vehicle between reception of appearance update packets from it. This requires storing the new position coordinates, velocity, orientation, and other appearance parameters in a RVA table entry for the remote vehicle.

Use of worldwide coordinates in the SIMNET protocol must not introduce a serious computation load at this point, either.

In summary, we must introduce a global coordinate system in the SIMNET network protocol without exacting a serious penalty in the computation required of participating simulators.

1.4 Requirement for Consistency of Position, Azimuth, and Range Calculations.

The network must supply position information at a precision in excess of the requirement of the most critical simulator or application. If the representation of positional information is based upon any model simpler than the best available (i.e., geodetic coordinates), then it will be inadequate whenever a simulation does not meet the restrictions of the model. For example, disseminating position information in a Universal Transverse Mercator (UTM) coordinate system will not work well when the simulation spans a large east-west range. Similarly, Universal Polar Stereographic (UPS) is not appropriate away from the poles, nor is the Lambert projection (projection onto a cone coaxial with the Earth) applicable for areas of large north-south extent.

Adequacy of a scheme for specifying and communicating position information depends on whether pairs of individual simulators can be made to agree as closely as desired on important geometric observations of the simulated world. The most important observations are position, azimuth, and range.

Agreement on position matters when an internal calculation of position within a simulator is compared to a position supplied via the network. If the external and internal positions are not consistent, that simulator and others on the network may not agree on the state of the world.

Likewise, azimuth and range inconsistencies will give rise to interoperability failures. The best mechanism to ensure consistency is to make the least restrictive assumptions about the nature of the space within which simulations are to take place. This proposal assumes only that activities take place within the general vicinity of the planet Earth.

2.0 BACKGROUND

2.1 The Defense Mapping Agency

The mission of the DoD Defense Mapping Agency (DMA) is to produce and distribute mapping, charting, and geodetic products to Department of Defense users worldwide. The DMA produces digital geographic information in a number of forms (Ref. 2). Two important products are Digital Terrain Elevation Data (DTED), which provides terrain altitude data at fenceposts spaced by 3 seconds of latitude and longitude (about 90 meter spacings) within a specified 1 degree by 1 degree geographic cell, and Digital Feature Analysis Data (DFAD), which provides digital description of linear features (e.g., roads, rivers, railroads) and area features (e.g., towns, lakes, tree canopies) within the same geographic cell.

2.2 The World Geodetic System 1984 (WGS84)

The use of grids, datums, and ellipsoids within the DoD is described in DMA Technical Manual 8358.1, Datums, Ellipsoids, Grids, and Reference Systems (Ref. 3). The DMA has chosen to refer all its products to a single coordinate system to support the widest range of worldwide applications, to easily relate information from different sources, and to ensure smooth transitions in product use from one part of the world to another.

The coordinate system chosen is the World Geodetic System, an Earth-centered (geocentric), Earth-fixed coordinate system that models the Earth gravimetrically (Ref. 6). This system uses extensive satellite tracking data to model the Earth's geoid, or gravitational equipotential surface, as an oblate spheriod (or ellipsoid, an ellipse of revolution). This ellipsoid approximates the undisturbed mean sea level worldwide. This model has undergone successive refinements, based on improved satellite data, in 1960, 1966, 1972, and 1984. The latest and most accurate version is called WGS84. Position coordinates within WGS84 consist of geographic coordinates (latitude and longitude) plus the altitude, with respect to the defining ellipsoid, as shown in Figure 2.

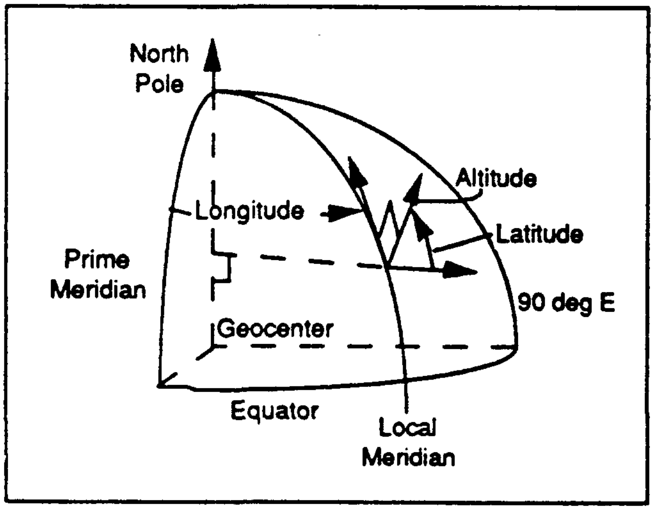


Figure 2. World Geodetic System 1984

For most applications, elevations and depths are measured with respect to a simplified form of the actual best approximation to the sea-level surface (the geoid). The geoid may deviate from the ellipsoid by several hundred meters. The practical consequence of these deviations is that the vertical reference used for depths and elevations is usually not the ellipsoid but an approximation to the geoid.

To further complicate matters, several vertical reference systems may apply to a single location on the Earth's surface. For example, a location on a beach may be referenced to mean lower low water on a nautical chart and to mean sea level on a land map. The elevations may differ by several meters. Practical vertical positioning requires some degree of universai agreement on a vertical reference surface, in addition to the ellipsoidal surface associated with a geodetic reference system.

This can be achieved with a relatively simple geoid model, perhaps even equating the geoid to the WGS84 ellipsoid, unless the most demanding,simulation application requires a more precise definition of the exact shape of the earth's surface. A standard geodetic reference system which provides Geodetic coordinates, in combination with the parameters that specify the associated ellipsoid, provide the most concise, universally accepted specification of location.

Simulations of Naval forces and Air Force units need to provide the same soldier-machine interfaces as the real platforms, namely input and output of coordinates with respect to WGS84.

The NAVSTAR Global Positioning System (GPS) is a constellation of navigation satellites, a number of monitoring and control ground stations, and mobile user equipment sets. The GPS provides highly precise position, velocity, and time information to users anywhere in the world, at any time, regardless of weather conditio . The GPS reports three dimensional geodetic positions in relation to WGS84, so this data can be directly related to other DMA map, charting, and geodesic products.

Many paper maps and other external data used in simulation exercises are based on geodetic reference systems other than WGS84. The Moledenskiy equations (or their abridged form, in Reference 3) may be used to convert between WGS84 coordinates and those based on other ellipsoids.

The Molodensky equations (Ref. 7) provide an accuracy of approximately 10 meters (Ref. 5). Other datum shift methods can provide accuracies approaching 15 centimeters in continental areas and 1 to 5 meters over the oceans. The precision (or internal consistency) of the transformation can be estimated by an analysis of the results of transforming a set of geodetic coordinates from a given ellipsoid to another and back again. An inverse transformation may not exactly reverse the effects of a transformation for two reasons: first, either of them may be an approximation, which has some error due to truncation of high order terms; second, finite numerical precision causes some loss of accuracy in each direction.

2.3 The Military Grid Reference System (MGRS)

The Military Grid Reference System (MGRS) is a method of projecting the surface of the Earth onto a series of 2-D maps ruled in parallel lines intersecting at right angles and forming a regular series of squares. The projections used are conformal, meaning that small geographic features retain their shape, intersection angles on the surface retain their true values, and, at any point, the scale factor is the same in all directions.

Military topographic maps are produced using the MGRS system and are used extensively by the Army for operations within a limited area. Coordinates on the grid are measured in meters (or kilometers) East and North of the map coordinate origin, so azimuths and ranges can be read directly from the map.

For latitudes between 80 degrees South and 84 degrees North, the projection used is called Universal Transverse Mercator (UTM). This projection may be viewed as placing a cylinder in contact with a selected meridian (the central meridian of the projection) on the ellipsoid representing the Earth's surface, and projecting the Earth's surface onto the cylinder from the center of the Earth, as shown in Figure 3. The axis of the cylinder is at 90 degrees from the Earth's axis, hence the term transverse.

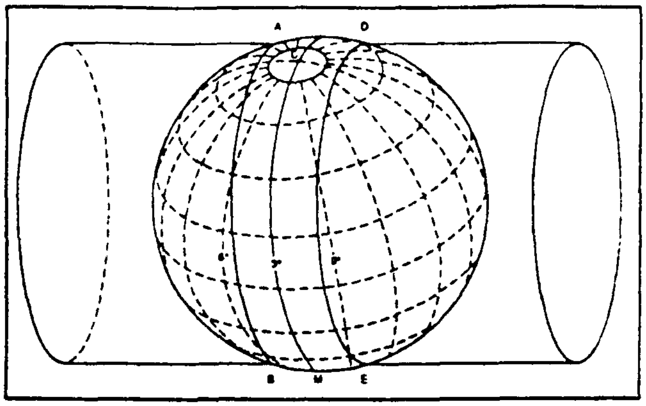


Figure 3. Universal Transverse Mercator Projection

Registration of the projection and the actual surface is perfect along the central meridian; to the East or West, the scale factor increases monotonically. To control this scale distortion, the projection width is limited to plus and minus 3 degrees from the central meridian, and the projection operation is repeated 60 times, with successive 6 degree slices of longitude, to cover the entire circumference of the Earth. (The cylinder is actually placed secant to the globe rather than tangent to minimize the deviation of the scale factor from 1.0 across the slice.)

For latitudes more than 80 degrees South and 84 degrees North, MGRS uses Universal Polar Stereographic (UPS) projections. For the North Polar Region, this may be viewed as placing a plane tangent to the North Pole and projecting the Earth's surface onto it from the South Pole, as shown in Figure 4. Conversely, for the South Polar Region, we project from the North Pole onto a plane tangent at the South Pole. Each of these projections is conformal. (Again, the plane is actually placed secant to the globe rather than tangent to minimize the deviation of the scale factor from 1.0 across the circular region.)

In simulations of Army platforms, the soldier-machine interfaces must support input and output of MGRS coordinates, as they do in the real vehicles.

Standard algorithms to convert in both directions between MGRS coordinates and WGS84 coordinates are given in Reference 4. As long as these conversions are only carried out to support soldier-machine interfaces (e.g., once per second), their computational efficiency is not a serious issue.

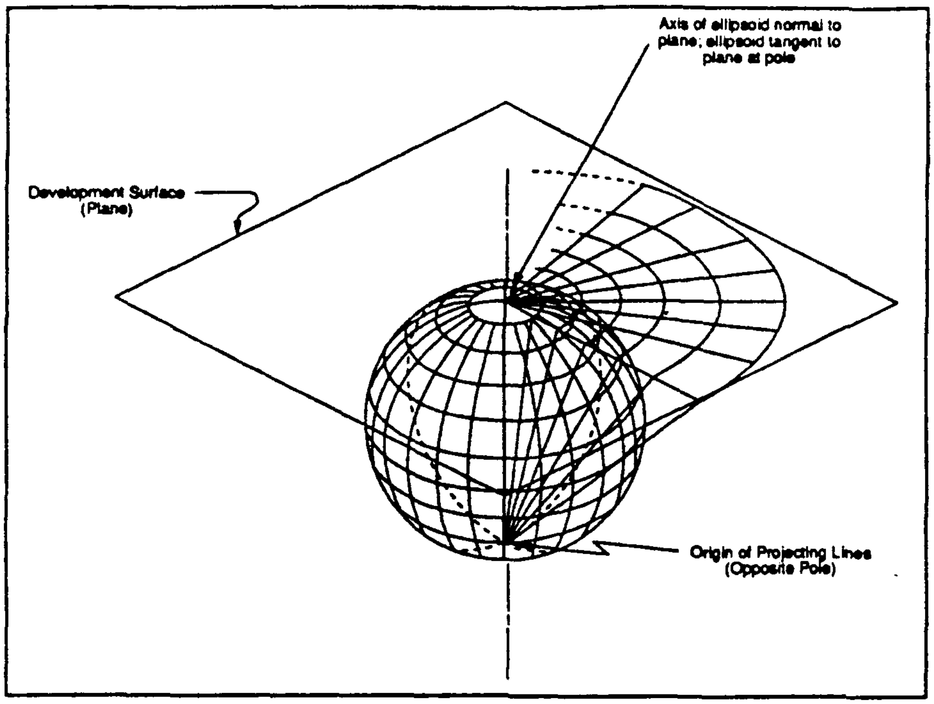


Figure 4. Universal Polar Stereographic (UPS) Projection

3.0 Assumptions

3.1 Cartesian Coordinates in Computer Image Generation Subsystem

We assume that each simulator's computer image generation (CIG) subsystem will operate in Cartesian coordinates. The positions of platforms, terrain, and cultural features to be displayed must be expressed in some x, y, z coordinate system to drive the CIG, and this coordinate system must extend out at least as far as the simulated platform's battlefield interaction range.

Since all reference to this coordinate system occurs internally to the simulator, it does not need to be standardized in the same way that a SIMNET network protocol does. Each simulator may keep its own Cartesian CIG coordinate system, converting between it and (standardized) global coordinates on the network, and each simulator's local coordinate system may be different from that of every other simulator. There are three obvious alternatives here for the choice of Cartesian CIG coordinates:

1. **Geocentric coordinates:** Earth geodetic centered, Earth-fixed coordinates aligned with the (Equator, Prime Meridian), (Equator, 90 degree E), and North Pole.
2. **Topocentric coordinates:** coordinates centered at a selected point on the Earth's surface and aligned at the selected point with East, North, and Up.
3. **Offset coordinates:** centered at some point on the Earth's surface, but aligned the same as Geocentric coordinates.

In any case, we require any conversion from the network standard global coordinate system to the CIG coordinate system to be efficient, because we may be carrying out this conversion 100 times per second when 100 other vehicles are visible. In the future, as high performance, low cost reduced instruction set computers (RISC machines) are introduced into most simulator implementations, far more arithmetic cycles will be available for less cost, so the computational efficiency of these coordinate transformations will become less of a serious issue.

3.2 Range Rejection Requirements

Our second assumption is that a simulator will need to efficiently reject about 1000 packets per second from 1000 other vehicles which are beyond interaction range. This assumption is based on the simulator data flow architecture shown in Figure 5.

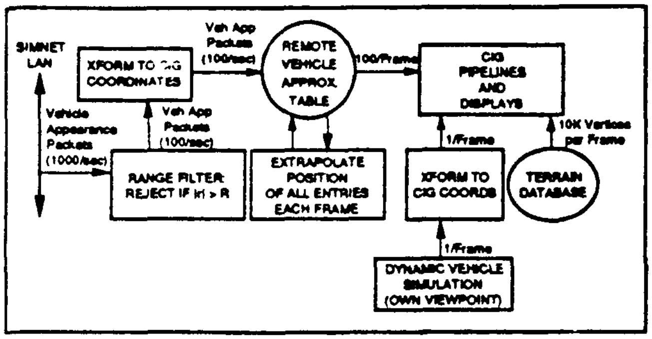


Figure 5. Data Flow Architecture

This restriction may be relaxed in the future by use of multicast addresses that are dynamically assigned to groups of mutual interest, i.e., only groups in which battlefield interactions are possible. Many local network implementations provide hardware filtering of multicast group addresses; so, in the future this burden may be removed from software executing in the simulator, and the processing burden of this range rejection computation may no longer be an issue.

3.3 Internal Simulator Simplifications and Optimizations Required

With current technology, it would be excessively expensive to require simulators to operate in real time, directly applying the full empirical and mathematical relationships between universal network coordinates, an ellipsoidal approximation to the Earth's surface, and the actual elevation or depth with respect to the geoid. We therefore assume that simulators and other network applications will make extensive simplifying assumptions about the representation of position and orientation with respect to the Earth. These engineering compromises may take any form whatsoever as long as global consistency in (at least) position, azimuth, and range is retained.

As an example, we might consider subpixel accuracy adequate for a visual system. A variety of simulators, from low-resolution, low-cost systems to high-end flight simulators may play together in an exercise. Implementation of subpixel accuracy would be very different for systems at opposite ends of the performance spectrum. Likewise, the requirements for a surface vehicle simulator, viewing horizontally, and a plan view display, viewing vertically, may differ in emphasis on preserving range versus azimuth.

4.0 The Key Issue: Choice of Global Geometry

4.1 The Globe

Maps normally project a region of the Earth's surface onto a plane ruled with a square grid. Although various projections may be specialized to display regions of large extent parallel to one of the two axes of the grid, none is satisfactory for covering a large region of the Earth's surface. A plane projection cannot reflect the geometric truth that the Earth's surface is curved. Curvature shows up in two equivalent ways:

1. **Position-dependent directions.** Topocentric directions change with position. The vectors East, North, and Up point in different directions from different points on the Earth's surface. On a spherical Earth, the angle between the Up vectors at two points is the great circle distance between the two points divided by R_e, the Earth's radius (about 6378 km).

2. **Angular Discrepancy.** Angles in a curved space (e.g., bearing differences on the Earth's surface) are different from those in a flat (plane) space. For instance, in a plane triangle, the sum of the three interior angles is always π radians. In contrast, the sum of the angles in a spherical triangle (made by arcs of three great circles on a spherical Earth) exceeds π by the curvature ($1/R_e^2$) times the area enclosed, as shown in Figure 6. The larger the area covered, the larger is the angular discrepancy. One result of this is that the back azimuth (bearing from a landmark) need not differ by 180 degrees from the azimuth (bearing to the landmark).

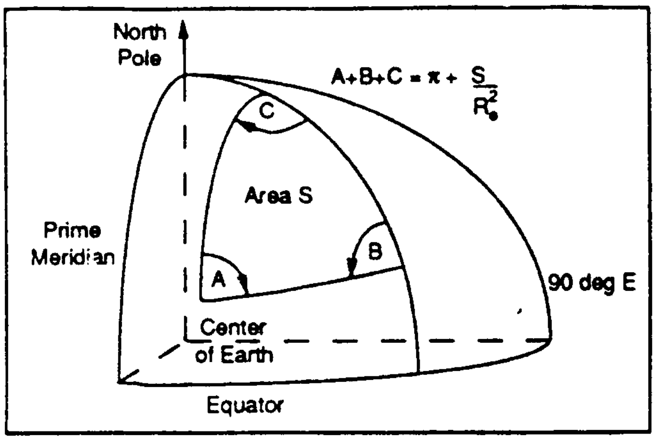


Figure 6. Angular Discrepancy on a Curved Surface

Either of these effects can be taken as the definition of curvature. However, the angular discrepancy generalizes more readily to higher dimensions.

In a global coordinate system, position-dependent directions require us to calculate the topocentric directions as a function of a simulator's current position. This can be reported to the dynamic simulation as a 3 by 3 orthonormal matrix containing East, North, and Up basis vectors as its columns. Equivalently, it is the orthonormal (rotation) matrix which multiplies a vector to convert it from topocentric coordinates to geocentric coordinates. Fortunately, this matrix does not change very often: if we recalculate it every time the simulator moves 6 kilometers, the Up axis will always be within a milliradian of the correct direction. (East and North directions change more rapidly and even undergo a 180 degree discontinuity when passing over the pole. This makes it desirable to recalculate the matrix more often.)

4.2 Projections

The idea of a projection is to reproduce the surface of the Earth on a plane such that the Up vector is a constant normal to the surface. Coordinates can then be taken which give two coordinates in the plane, with altitude above the plane representing altitude above mean sea level. The constant Up vector may prove convenient in the dynamic vehicle simulation and in ballistics simulation.

However, due to the angular discrepancy effect, distortions inevitably result from projecting a curved surface onto a plane. These distortions may take the form of angle distortions (giving different angles in the plane from those on the Earth's surface), distance distortions (changes in scale factor from place to place) or both. Equivalently, we can map one set of great circles into straight lines in the plane, but we cannot do this with all great circles.

For example, the Mercator, Lambert conic, and UPS projections may be viewed as rolling up a plane along an axis coincident with the Earth's axis, placing the resulting surface (cylinder, cone, or plane, respectively) tangent to the Earth, and transferring geographic features onto the adjacent surface. These reproduce meridians as straight lines, but other great circles appear as circular arcs on the map (except the Equator which appears as a straight line in the Mercator projection).

The Universal Transverse Mercator (UTM) projection is onto a cylinder with axis rotated 90 degrees from the Earth's axis, and tangent to the Earth along a central meridian. Great circles that intersect the central meridian at 90 degrees map into straight lines, as does the central meridian, but other great circles map into circular arcs.

We cannot eliminate these angular distortions because they are inherent in any projection of a curved geometry onto a plane. The best we can do is to require the projection to be conformal. This means that intersection angles are preserved in the mapping process, or equivalently, the scale factor of the map at a single point is the same in all directions. Small geographic features are orthomorphic (same shape) but they are scaled and rotated by different amounts at different locations. For the projections above, the scale factor is minimum at the center of the resulting map and increases toward the periphery.

4.3 Drawbacks of Projections

The use of a projection to represent global geometry has four major drawbacks:

1. **Flat Earth Approximation.** All these projections represent the Earth as flat. This means that a CIG driven by a projected geometry will be unable to show the curvature of the Earth's horizon, even from high altitude vehicles. This is not a significant effect

for SIMNET vehicles, though it may be a problem in space shuttle and National Aerospace Plane simulators. More serious is the need in naval simulations to duplicate the effect of incoming sea-skimming missiles popping up over the horizon, or other ships sinking below the horizon as they steam away, breaking optical communications links.

2. **Runtime Projection Costs.** All terrain and all moving models must have their geometry (e.g., vertex locations) transformed by exactly the same projection. If moving models were transformed differently than terrain, a vehicle hidden behind some terrain feature (e.g., a small hill) could appear in some viewing CIG as appearing in front of the hill, or inside it, or floating above it. This imposes a serious runtime penalty: the appearance packets arriving from the 100 or so other vehicles within viewing range (about 100 packets per second) must be transformed through the projection geometry without consuming a serious fraction of the simulator's computing resources.

   In order to efficiently approximate one of the projections above within less than a meter over 10 kilometer ranges, we generally need a second order approximation, i.e., a quadratic function of two variables requiring over of a dozen arithmetic operations per appearance packet. The truncation error introduced in this approximation (neglecting higher order terms) adds to the distortion errors introduced by the projection transformation itself.

3. **Bearing and Range Distortions.** These effects, resulting from projecting a curved surface onto a flat one, were described above. One way that a surveyor might measure them within a simulated environment is by use of navigation equipment aboard a simulated Apache helicopter. A list of exact locations of a number of significant landmarks are stored in the digital navigation computer. The TADS (target acquisition and designation system) can be used to obtain a fix by laser ranging off a known landmark. The measured range and bearing to the landmark, plus its stored location, give the vehicle's position. If the Apache is located at some other landmark and there is significant range or bearing distortion, it will be impossible to reconcile this measured location with this landmark's stored location.

4. **Reprojecting all Visible Terrain.** The projection distortions described above grow quadratically with the size of the region being projected. In order to keep distortions within reason, we must keep the size of the region very small with respect to the Earth's radius. As the simulated vehicle moves across the Earth, it will eventually reach a point where the distortion at the edge of the projection is all we can tolerate. At this point, we must switch to a new projection, centered near our current location. Switching to a new projection means recomputing the projection geometry for every vertex in the terrain within current view. To prevent a distracting flash on the CIG, this replacement of the visible terrain should be done within one video frame time. This requires either double buffering the terrain or providing a very fast recomputation cycle to reproject the 10,000 or so vertices within range.

4.4 Use of Geocentric Cartesian Coordinates

Because of these drawbacks to the use of projective geometry, we instead recommend the use of Cartesian geocentric coordinates in the vehicle appearance packets. Since we are not performing any projection, there are no distortions of bearings or ranges; angles and lengths are preserved exactly. We are not forcing the Earth to be flat. A periodic reprojection of all the terrain in range is not needed. These Geocentric Cartesian coordinates are shown in Figure 7.

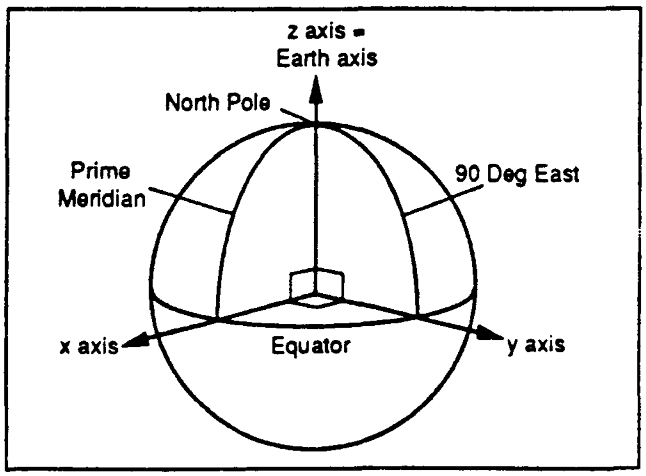


Figure 7. Geocentric Cartesian Coordinates

In order to do the range filtering of packets from vehicles beyond interaction range, we replace the appearance packet rejection test of Section 1.3 with:

Reject if:

$$x < x_0 - R \text{ or } x > x_0 + R \text{ or } y < y_0 - R \text{ or } y > y_0 + R \text{ or } z < z_0 - R \text{ or } z > z_0 + R$$

This results in six arithmetic operations (comparisons with precomputed boundaries) instead of four, or about 6,000 arithmetic operations/second.

The small minority of packets that pass this filter can be subjected to a subsequent tighter range test, namely:

Reject if:

$$(x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2 \geq R^2$$

This requires an additional nine arithmetic operations for the packets which passed the first test, or about 900 operations/second more.

If the CIG also operates in Cartesian geocentric coordinates, we can choose to do the RVA (remote vehicle approximation) in Cartesian geocentric coordinates as well, so there is no runtime penalty in terms of arithmetic operations to perform a transformation. Of course, this is a design choice to be made by a simulator vendor, not a specified standard. We discuss it here only to demonstrate the existence of a design choice that minimizes the added computational burden from use of global coordinates.

The choice of Cartesian geocentric coordinates for a global coordinate system meets our stated objectives for converting the SIMNET network protocol to a global coordinate system without imposing unacceptable runtime burdens on the simulators. However, it does raise three new issues:

1. **Coordinate Resolution.** The current SIMNET network protocol uses 64-bit floating point IEEE Standard 754 values for x, y, and z coordinates. We will now interpret x, y, and z as geocentric Cartesian coordinates. The 64-bit numbers will range between about ± 6.4 million meters with 52-bit mantissa for a sub-micron resolution worldwide. This resolution is far greater than we need for simulation applications. On the other hand, 32-bit floating point would only give about 1 meter resolution, which would result in visibly jerky movement of nearby vehicles.

   A simulator vendor could choose to take advantage of the fact that we do not need an entire 64 bits of resolution within the CIG by adopting offset coordinates for the CIG (aligned with geocentric Cartesian directions but centered at a point on the Earth's surface). Each region of terrain will then have a 64-bit origin x, y, z stored with it, and vertex values within the terrain region will be stored as 32-bit floating point differences from that origin. To interpret a position within a vehicle appearance packet, we subtract the vehicle position from our current origin (in 64-bit precision), then round the difference to 32-bit floating point. This position is then commensurate with the 32-bit floating point values of terrain vertex locations, and both will feed a 32-bit CIG. This 32-bit resolution (23-bit mantissa) gives a resolution of 1 centimeter over an 80 kilometer distance from the origin of the terrain region.

   The runtime implication of this design choice is three subtractions per vehicle appearance packet before storing the result in the RVA table, or about 300 additional arithmetic operations per second.

2. **Choice of Coordinates for the Simulation Host.** A single simulation host is likely to use a number of different coordinate systems. For example, calculation of aerodynamic forces is done most economically in relative wind coordinates, where drag is parallel to relative wind direction and lift is orthogonal to it. Calculation of angular acceleration is done most economically in body coordinates, where the tensor of inertia is a constant matrix. Display of heading and attitude is done in topocentric coordinates (East, North, Up) as shown in Figure 8. Integration of angular velocity into orientation may be done most economically in a quaternion representation (a generalization of complex numbers), which may be with respect to either geocentric or topocentric coordinates. Integration of acceleration into velocity and velocity into position may be done equally easily in either geocentric or topocentric coordinates.

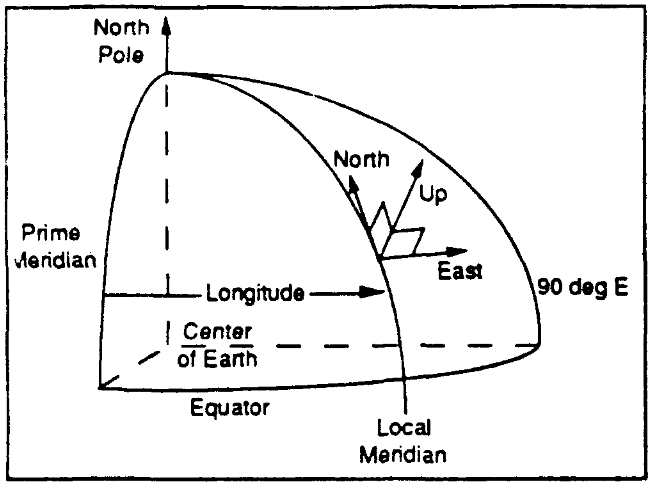


Figure 8. Topocentric Coordinates

We assume that the CIG's database processor will supply the dynamic vehicle simulation with the rotation matrix and local origin needed to convert between topocentric and geocentric coordinates, and will update these values occasionally as appropriate. Algorithms for performing these conversions are given in Appendix A. To the extent the vehicle simulation calculates position, velocity, and orientation in topocentric coordinates, it must convert these to geocentric coordinates before reporting them to other simulators via the SIMNET network protocol, or to a CIG that uses geocentric coordinates. This conversion requires 18 arithmetic operations for every three-vector to send out an update. To the extent the vehicle simulation chooses to calculate position, velocity, and orientation in geocentric coordinates, none of these conversions are necessary.

3. **Conversions to and from WGS84.** The vehicle simulation can produce its position in geocentric coordinates as described above. However, Naval and Air Force soldier-machine interfaces will require input and output of geographic coordinates relative to WGS84, plus altitude above sea level. The algorithms for accomplishing these conversions are given in Appendix A. Since these conversions only happen at human interface speeds (e.g., once per second) their runtime efficiency is not a serious issue.

   Further conversion between WGS84 and MGRS to support Army platform soldier-machine interfaces may be performed using the algorithms in Reference 4.

5.0 Conclusions and Recommendations

We recommend adoption of a Cartesian geocentric coordinate system to represent positions, velocities, and orientations within the SIMNET network protocol.

Using the approaches described above, we achieve positional accuracies of a fraction of a meter worldwide, without incurring a significant increase in the runtime arithmetic operations required within a simulator.

6.0 References

1. Pope, Arthur, January 1990 Revision, The SIMNET Network and Protocols, BBN Report No. 7102, BBN Systems and Technologies Corporation, Cambridge, MA.

2. Digitizing the Future, DMA Stock No. DDIPDIGITALPAC.

3. Datums, Ellipsoids, Grids, and Reference Systems. DMA Technical Manual 8358.1, preliminary edition, Hager, Fry, Jacks and Hill.

4. Universal Transverse Mercator Grid, Department of the Army Technical Manual TM-5-241-8, April 1973.

5. Dewhurst, W.T., 1989, The Accurate and Rapid Transformation of Large Quantities of Positional Data from the North American Datum of 1983: *Proceedings of the GIS/LIS-89 Conference*, Orlando, FL, pages 69-81.

6. DMA 1987, DoD World Geodetic System 1984, Its Definition and Relationships with Local Geodetic Systems, DMA Technical Report TR 8350.2, 49 pages.

7. Molodensky, M.S., V.F. Eremeev and M.I. Yurkina, 1962, Methods for Study of the External Gravitational Field and Figure of the Earth, Israel Program for Scientific Translations.

APPENDIX A: COORDINATE CONVERSION ALGORITHMS

In this section we present a complete set of algorithms for interconverting between the various coordinate systems described in our proposal.

Definitions

The WGS84 ellipsoid is an ellipsoid of revolution defined by two parameters: the equatorial radius a = 6,378,137 meters (the semimajor axis of the ellipse); and the flattening f = 1 / 298.25722 3563. If we denote the polar radius (the semiminor axis of the ellipse) as b, then b = a (1 - f).

On the Earth's surface, geodetic/geographic longitude is the angle λ between the plane of the local meridian and that of the Greenwich meridian, with positive values to the East. Latitude ϕ is the angle between the local normal to the ellipsoid (local vertical) and the equatorial plane, with positive values to the North.

Three surfaces are important for measurement of vertical position in a geodetic system: the *reference ellipsoid*; the *geoid*, an idealized equipotential surface coinciding with mean sea level over the oceans and extending across (usually below) land masses; and the *physical surface* at the base of the atmosphere. They give rise to three related measures of height.

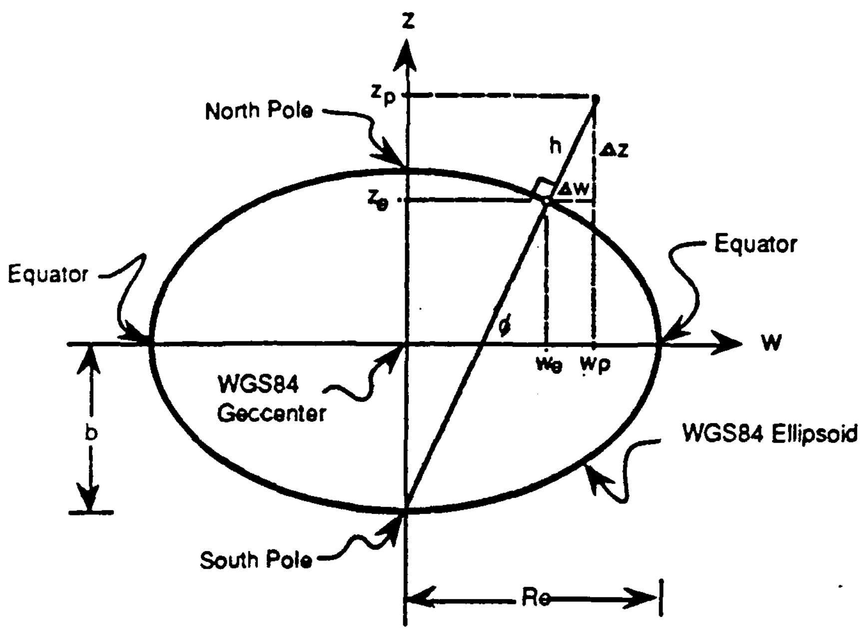
Geodetic height (H) is the altitude above or depth below the reference ellipsoid. Geoid height (N) is the altitude above or depth below the ellipsoid of the geoid. Surface height (elevation) is the elevation of the physical surface (h) with respect to the geoid.

Our Euclidian geocentric x, y, and z axes point from the center of the earth toward the (Equator, Greenwich meridian), (Equator, 90 degrees East longitude), and the North Pole respectively. Our z axis is aligned with the Earth's axis.

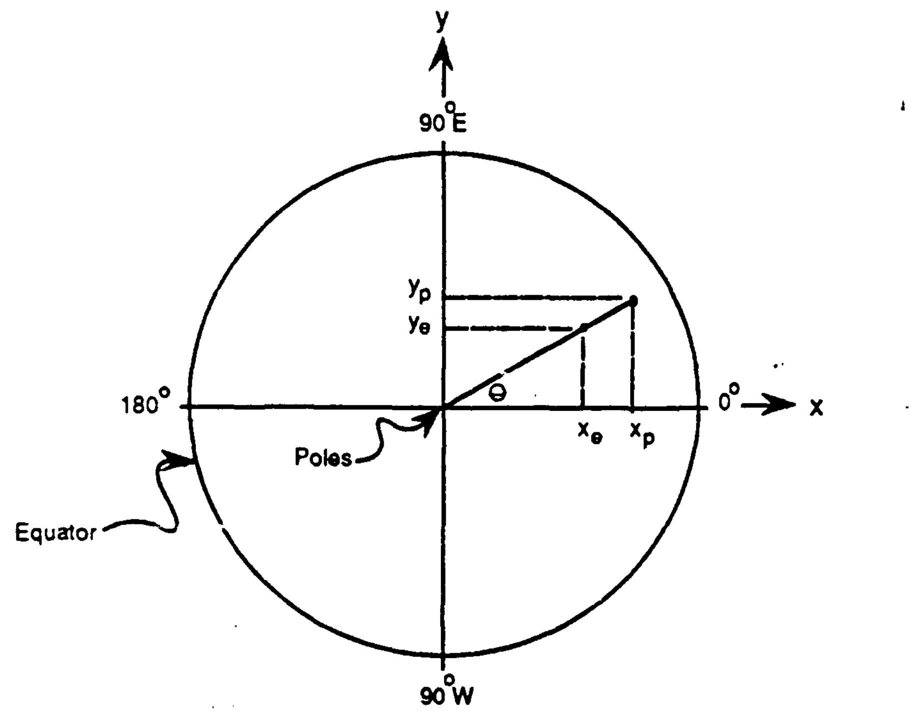
We denote distance from the Earth's axis as $w = \sqrt{x^2 + y^2}$. The defining equation for the generating ellipse is then:

$$\frac{w^2}{a^2} + \frac{z^2}{a^2(1 - f)^2} = 1$$

Rotating this ellipse around the z axis generates the WGS84 ellipsoid of revolution. Other ellipsoids may be generated using different values of the parameters a and f.



WGS84 Ellipsoid: Meridional Section



WGS84 Ellipsoid: Equatorial Section

Figure A-1. Cross Sections of a Reference Ellipsoid

Interconversion Algorithms

The following algorithms are expressed in the *Pascal* programming language. Brief derivations of the mathematical basis of the more complicated procedures supplement the *Pascal* implementation. A count of the trigonometric function calls, square root function calls, and floating point operations is given with each statement and as a summary at the end of each function or procedure. A short analysis of the floating point operations follows each algorithm. These algorithms are accurate to within a centimeter assuming their application within several kilometers of the earth's surface and 64-bit IEEE floating point arithmetic.

The interconversion algorithms are presented with a bias toward clear definition of their function. While they are reasonably economical as presented, the reader will undoubtedly see modifications that could be made to reduce their computational (time) cost. In particular, it may be desirable to combine several procedures where they will normally be called in sequence. While such improvements are useful in a practical sense, the numbers presented fairly represent magnitude of the various conversion procedures.

The basic unit of time is the 64-bit floating point operation. This unit is designated as a *float*. The square root operation requires a *sqrt* of time to compute. Trigonometric functions require a *trig*. A *sqrt* was found to equal 5 *floats*, and a *trig* was found to equal 12 *floats* in our computing environment (Sun 3; C language). All other operations (including floating point assignment and sign reversal of a floating point value) were considered to be of negligible computational cost.

Whenever the term "ellipsoid" is used without qualification, the WGS84 reference ellipsoid is intended.

Interconversion between position on geoid and ellipsoid vertical position is most easily determined with respect to sea level, or its approximating surface, the geoid. Simulation of vehicles, sensors, and weapons operating over intercontinental ranges or from orbital altitudes may require accurate vertical position determination. Current Simnet applications are not sensitive to deviations of the geoid from the ellipsoid and the approximation that the ellipsoid and geoid coincide is quite reasonable.

The geoid could be modeled using some representation of the vertical separation of the two surfaces as a function of horizontal position. The simplest function is a constant (perhaps 0.0). This model is adequate for simulations up to continental scale. More precise (and computationally expensive) models based on polynomial expansions or interpolation of tabulated values could be used for precise global

simulations. Note that such precision would not be required for global simulations unless some sensor, weapons system, or navigational exercise required an accuracy of less than about 100m over the entire planet.

Global Variables

The conversion algorithms that follow share two sets of global variables: parameters that define the ellipsoid used by the geodetic system used or which are derived from those defining parameters, and elements of the matrices used in performing the topocentric conversions. These variables are initialized by corresponding procedures: initialize_ellipsoid sets up the ellipsoid variables, and initialize_topocentric sets up the topocentric conversion variables.

```
var
        ellipse_a: real;                { Equatorial radius }
        ellipse_a_sq: real;             { a squared }
        ellipse_f: real;                { Flattening: (a-b)/a }
        ellipse_b: real;                { Polar semi-diameter }
        ellipse_b_sq: real;             { b squared }
        ellipse_a_sq_over_b: real;      { a*a / b }
        ellipse_e: real;                { Ellipticity: 2f-f*f }
        ellipse_c1: real;               { (1-f)*(1-f) }

        topo_radius: real;      { z value at origin of topocentric system }
        topo_sin_lat: real;     { sine of angle between normal and equatorial plane }
        topo_cos_lat: real;     { cosine of angle between normal and equatorial plane }
        topo_sin_lon: real;     { sine of angle between normal and plane of prime meridian }
        topo_cos_lon: real;     { cosine of angle between normal and plane of prime meridian }
        topo_sin_cos: real;     { sin(lat)*cos(lon) }
        topo_sin_sin: real;     { sin(lat)*sin(lon) }
        topo_cos_cos: real;     { cos(lat)*cos(lon) }
        topo_cos_sin: real;     { cos(lat)*sin(lon) }
{
This procedure sets up constants associated with a reference ellipsoid associated with a geodetic system.
The two input parameters, a and f, are the equatorial radius and the flattening, respectively. By
convention the equatorial radius is given in meters. For WGS84, a is 6,378,137m and f is
1/298.257223563. It must be called once before performing any conversions or invoking the
initialize_topocentric procedure. It may be called with new parameters to allow conversions with respect to
a different ellipsoid.
}

procedure initialize_ellipsoid(a, f:real);
        begin
```

```
        ellipse_a := a;                                               { no op }
        ellipse_a_sq := ellipse_a*ellipse_a;                          { 1 float }
        ellipse_f := f;                                               { no op }
        ellipse_b := ellipse_a*(1.0-ellipse_f);                       { 2 float }
        ellipse_b_sq := ellipse_b*ellipse_b;                          { 2 float }
        ellipse_a_sq_over_b := ellipse_a_sq/ellipse_b;                { 1 float }
        ellipse_e := ellipse_f*(2.0+ellipse_f);                       { 2 float }
        ellipse_c1 := (1.0 - ellipse_f) * (1.0 - ellipse_f);          { 3 float }
    end; { initialize_ellipsoid }                                     { 11 float }
```

The computational cost of this procedure is 11 floating point operations.

```
{
This procedure sets up transformation constants associated with a point with geocentric coordinates x, y,
and z for use in subsequent conversions between geocentric and topocentric coordinates. It must be
called once whenever a new origin is selected for a topocentric coordinate system.
}
procedure initialize_topocentric( x, y, z: real)
    var
        w, wsq, lat, lon, dummy: real;
    begin
        wsq := x * x + y * y;                                         { 3 float }
        topo_radius := sqrt(wsq + z * z);                             { 1 sqrt; 2 float }
        geocentric_to_geodetic(x, y, z, lat, lon, dummy);             { 85 float }
        w := sqrt(wsq);                                               { 1 sqrt }
        topo_cos_lat := cos(lat);                                     { 1 trig }
        topo_sin_lat := sqrt(1.0 - topo_cos_lat * topo_cos_lat);      { 1 sqrt; 2 float }
        topo_cos_lon:= x / w;                                         { 1 float }
        topo_sin_lon := y / w;                                        { 1 float }
        topo_sin_sin := topo_sin_lat * topo_sin_lon;                  { 1 float }
        topo_sin_cos := topo_sin_lat * topo_cos_lon;                  { 1 float }
        topo_cos_sin := topo_cos_lat * topo_sin_lon;                  { 1 float }
        topo_cos_cos := topo_cos_lat * topo_cos_lon;                  { 1 float }
    end; { initialize_topocentric }                                   { 1 trig; 3 sqrt; 98 float }
```

The total computational cost of this procedure is approximately 123 floating point operations.

Interconversion between Geodetic and Geocentric

The conversion from geodetic coordinates to geocentric coordinates involves locating a point at height h above an ellipsoidal surface at a point where the surface normal is equal to the tangent of the geodetic latitude.

The requirement is to compute the coordinates of a point x_p, y_p, z_p at the given height, h, above a point on the ellipsoidal surface at latitude ϕ, longitude λ.

Let $w(x, y) = \sqrt{x^2+y^2}$ be a horizontal coordinate in a meridional section (Figure A-1).

Then

$$x_p = w_p \cos \lambda \qquad \text{(A-1)}$$

and

$$y_p = w_p \sin \lambda \qquad \text{(A-2)}$$

The defining equation of the ellipse in the cross section is

$$\frac{w^2}{a^2} + \frac{z^2}{b^2} = 1 \qquad \text{(A-3)}$$

where a is the equatorial radius, b the polar semi-diameter, and

$$f = \frac{a}{a - b} \qquad \text{(A-4)}$$

Solving (A-3) for w gives

$$w_e = a\sqrt{1 - \frac{z^2{}_e}{b^2}} \qquad \text{(A-5)}$$

where the subscript e indicates a point on the ellipsoid.

It follows that

$$\cot \phi = \frac{dz}{dw} = \frac{b}{a^2} \frac{w}{\sqrt{1 - \frac{w^2}{a^2}}} \qquad \text{(A-6)}$$

and

$$w_e = \frac{a \cos \phi}{\sqrt{1 - e \sin^2 \phi}} \qquad \text{(A-7)}$$

where $e = 2f-f^2$.

Then

$$z_e = a(1 - f)\sqrt{1 - \frac{w^2{}_e}{a^2}} \qquad \text{(A-8)}$$

and

$$w_p = w_e + h \cos \phi \qquad \text{(A-9)}$$

and

$$z_p = z_e + h \sin \phi \qquad \text{(A-10)}$$

Finally

$$x_p = w_p \cos \lambda \qquad \text{(A-11)}$$

and

$$y_p = w_p \sin \lambda. \tag{A-12}$$

```
{
This procedure converts a position specified by the geodetic coordinates lat (latitude), lon (longitude),
and height (height) to Cartesian geocentric coordinates x, y, and z. Initialize_ellipsoid must have been
called prior to using geodetic_to_geocentric.
}
procedure geodetic_to_geocentric( lat, lon, height: real; var x, y, z: real);
    var
        sinlat, temp1, temp2, w;

{ The algorithm is based directly on the mathematical derivation given above }

    begin

    { Obtain the vertical and horizontal projection of the point on the ellipsoid }

        sinlat := sin(lat);                                          { 1 trig }
        temp1 := ellipse_a / sqrt(1.0 - ellipse_e * (sinlat * sinlat));  { 1 sqrt; 4 float }
        temp2 := temp1 * ellipse_c1;                                 { 1 float }
        temp1 := temp1 + height;                                     { 1 float }
        temp2 := temp2 + height;                                     { 1 float }

    { Obtain the projected horizontal position on the equatorial plane }

        w := temp1 * cos(lat);                                       { 1 trig; 1 float }

    { Obtain the projected vertical position on the polar axis }

        z := temp2 * sinlat;                                         { 1 float }

    { Project the horizontal position on the two axes in the equatorial plane }

        x := w * cos(lon);                                           { 1 trig; 1 float }
        y := sqrt(w * w - x * x);                                    { 1 sqrt; 3 float }

    end  { geodetic_to_geocentric }                                  { 3 trig; 2 sqrt; 13 float }
```

The total computational cost of this procedure is approximately 59 floating point operations.

The conversion from geocentric coordinates to geodetic coordinates involves locating a point on an ellipsoidal surface that is directly below a given point. This conversion is potentially less straightforward than the previous one since it is possible to locate as many as four points on an ellipsoid that meet the stated condition. Normally the desired solution is the nearest point that meets the conditions. Although it is possible to compute the solutions directly and determine the correct one, an iterative procedure is computationally less expensive.

In outline, the algorithm used here is based upon finding a point on the ellipsoid that is reasonably close to the point below the given point. The slope of the ellipsoid is used to estimate the location of the solution with respect to the initial guess, and the guess is updated. The process is repeated until the distance between the current estimate and the next estimate is less than some tolerance (one meter is used in the *Pascal* version). For given point within a few kilometers of the ellipsoid, a single iteration results in accuracy within a meter.

The aim is to calculate the height, h, above the ellipsoid and the latitude ϕ, and longitude λ of a point whose Cartesian geocentric coordinates are x_p, y_p, and z_p. Again, let $w(x, y) = \sqrt{x^2+y^2}$ be a horizontal coordinate in a meridional section.

The first step is to compute the horizontal coordinate

$$w_p = \sqrt{x^2 + y^2} \qquad \text{(A-13)}$$

and locate a point on the ellipsoid

$$w_e = \frac{w_p}{\sqrt{\dfrac{w^2_p}{a^2} + \dfrac{z^2_p}{b^2}}} \qquad \text{(A-14)}$$

the slope at w_e is

$$\tan \phi = \frac{a^2}{b}\sqrt{1 - \frac{w^2_e}{a^2}} \qquad \text{(A-15)}$$

The magnitude of the surface normal is

$$dn = \sqrt{1 - \frac{1}{\tan^2 \phi}} \tag{A-16}$$

and the vertical coordinate

$$z_e = b\sqrt{1 - \frac{w^2{}_e}{a^2}} \tag{A-17}$$

Let $dr_w = w_p - w_e$ and $dr_z = z_p - z_e$. Then

$$dr = \sqrt{dr^2{}_w + dr^2{}_z} \tag{A-18}$$

and h can be estimated as

$$h = \frac{dr_w + \frac{dr_z}{\tan^2 \phi}}{dn} \tag{A-19}$$

Then the arc distance from (w_e, z_e) to a location on the ellipsoid below (w_p, z_p) is approximately

$$ds = \sqrt{dr^2 + h^2} \tag{A-20}$$

The increment along the w direction is

$$dw = \frac{ds}{dn} \tag{A-21}$$

and a new estimate for w_e is

$$w_e[i+1] = w_e[i] + dw \tag{A-22}$$

Equations (A-15) through (A-22) can be repeatedly applied until the correction dw in (A-22) is sufficiently small.

```
{
This procedure converts a Cartesian geocentric position specified by the coordinates x, y, and z, to a
geodetic position lat (latitude), lon (longitude), and height (height). The procedure initialize_ellipsoid
must be called prior to using geocentric_to_geodetic.
}
procedure geocentric_to_geodetic( xp, yp, zp: real; var lat, lon, height);
    var
        wp, we, wesq, ze, zpsq, wpsq, h: real;
        tanphi, inv_tanphi_sq: real;
        dn, drw, drz, dr, ds, ds_sq, dw: real;
    begin

{ Compute a starting point at the intersection of a geocentric radius and the ellipsoid.
Working in the plane of the polar axis and the given point reduces the problem
temporarily to two coordinates, w and z }

        zpsq := zp * zp;                                        { 1 float }
        wpsq := x * x + y * y;                                  { 3 float }
        wp := sqrt(wpsq);                                       { 1 sqrt }
        we := wp /
            sqrt(wpsq / ellipse_a_sq + zpsq / ellipse_b_sq);    { 1 sqrt; 4 float }
        repeat                                                  { loop: 5 sqrt; 26 float }

    { Move along a meridian of the ellipsoid until the given point is on the surface normal }

            begin
                wesq := we * we;                                { 1 float }

    { Get current slope }
                tanphi := (ellipse_a_sq_over_b *
                    sqrt(1.0 - wesq/ellipse_a_sq)) /
                    we;                                         { 1 sqrt; 4 float }
                inv_tanphi_sq := 1.0 / (tanphi*tanphi);         { 2 float }

    { Get magnitude of normal }

                dn := sqrt(1.0+inv_tanphi_sq);                  { 1 sqrt; 1 float }
                ze := ellipse_b *
                    sqrt(1.0-wesq/ellipse_a_sq);                { 1 sqrt; 3 float }
                drw := wp - we;                                 { 1 float }
                drz := zp - ze;                                 { 1 float }

    { Compute distance from current location on ellipsoid to given point }
```

```
                    dr := sqrt(drw*drw + drz * drz);            { 1 sqrt; 3 float }

    { Estimate true height }

                    h := (drw + drz *inv_tanphi_sq) / dn;       { 3 float }

    { Estimate arc distance to point that would be under given point }

                    dssq := (dr * dr + h * h);                  { 3 float }
                    if dssq > 0.0 then                          { 1 float }
                        begin
                            ds := sqrt(dssq);                   { 1 sqrt }
                            dw := ds/dn;                        { 1 float }

    { Update position to new estimate }

                            we := we + dw ;                     { 1 float }
                        end;
                    else dw := 0.0;                             { no op }
                end;

    { Finished when last move is less than a meter }

            until dw < 1.0;                                     { 1 float }

    { Compute geodetic latitude }

            lat := arctan(tanphi);                              { 1 trig }

    { Compute longitude from components in equatorial plane }

            lon := arctan2(yp, xp);                             { 1 trig }
            height := h;                                        { no op }
    end     { geocentric_to_geodetic }          { 2 trig; 2+5*loop sqrt; 4+26*loop float }
```

The total computational cost of this procedure is approximately 34 floating point operations plus 51 floating point operations per iteration. Within several kilometers of the earth's surface the loop is executed only once, for a total of approximately 85 floating point operations.

Interconversion between Geocentric and Topocentric

This conversion is based on a rotation about three axes so that the z axis is oriented parallel to the surface normal, x is oriented toward the north pole and y is oriented toward the east. Following rotation, the coordinate system is translated along its z axis to the origin of the topocentric system.

```
{
This procedure converts between a Cartesian geocentric position xg, yg, and zg and a Cartesian
topocentric position xt, yt, and zt. The procedure initialize_topocentric must have been called at least
once prior to invoking geocentric_to_topocentric.
}
procedure geocentric_to_topocentric( xg, yg, zg: real; var xt, yt, zt: real);

{ Rotate the coordinates using the stored matrix elements }

    begin
        xt := xg * ( - topo_sin_lon) +
              yg * topo_cos_lon;                          { 3 float }
        yt : = xg * ( - topo_sin_cos) -
              yg * topo_sin_sin +
              zg * topo_cos_lat;                          { 5 float }
        zt := xg * topo_cos_cos +
              yg * topo_cos_sin +
              zg * topo_sin_lat -

    { Translate to the topocentric origin }

              topo_radius;                                { 6 float }
    end   { geocentric_to_topocentric }                  { 14 float }
```

The total computational cost of this procedure is 14 floating point operations.

This conversion is based on a translation along its z axis to the geocenter followed by an inverse of the geocentric_to_topocentric rotation about three axes.

```
{
This procedure converts between a Cartesian topocentric position xt, yt, and zt and a Cartesian
geocentric position xg, yg, and zg. The initialize_topocentric procedure must have ben called prior to
invoking topocentric_to_geocentric.
}
procedure topocentric_to_geocentric( xt, yt,zt: real; var xg, yg, zg: real);
    var
        tzt: real;
```

```
    begin

{ Translate to geocenter }

            tzt := zt + topo_radius;                         { 1 float }

{ Use transpose of rotation matrix to perform inverse rotation }

            xg := xt * ( - topo_sin_lon) -
                    t * topo_sin_cos +
                    tzt * topo_cos_cos;                      { 5 float }
            yg := xt * topo_cos_lon -
                    yt * topo_sin_sin +
                    tzt * topo_cos_sin;                      { 5 float }
            zg :=
                    yt * topo_cos_lat+
                    tzt* topo_sin_lat;                       { 3 float }
    end     { topocentric_to_geocentric }                    { 14 float }
```

The total computational cost of this procedure is 14 floating point operations.

Position Paper: Information Requirements for Unmanned Forces

Thomas I. Garvey, Principal Scientist
Artificial Intelligence Center
SRI International
333 Ravenswood Avenue.
Menlo Park, California 94025
(415) 859-3486; E-mail: Garvey@ai.sri.com

January 11, 1990

1 Introduction

Standards defined today will either enhance or impede the potential future utilization of systems such as SIMNET for activities such as the definition of new doctrine, requirements definition for new battlefield systems and support functions, and the support for research in decision aids and autonomous systems. An effective set of data standards will enhance the ability of "third party vendors" to provide useful software for direct support of simulation functions, as well as contribute to future requirements definition and evaluation.

Many requirements, such as standards for rendering and visualization processes, are already being addressed. The desire to introduce a new generation of sophisticated, simulated, unmanned forces, however poses new, unique requirements for data availability and data interchange processes which we outline briefly here.

2 Operational Requirements

The unmanned force elements will, ideally, operate in a manner similar to human tank crews. It should not be possible to identify an unmanned vehicle on the basis of its behavior, nor should unmanned forces be any less effective or deadly than their human counterparts. Human commanders should be

able to give orders to their unmanned forces in the same fashion that they would to manned vehicles under their command. The commander should also be able to anticipate accurately the likely behavior of their unmanned forces: it has been said that to a pilot or a tank commander, there are no pleasant surprises.

The "seamless" integration of manned and unmanned forces implies a number of capabilities for the unmanned forces. They must be able, in effect, to mimic the cognitive capabilities of their human counterparts. In particular, they must be able to perceive their environment, update and maintain a model of the developing tactical situation, plan actions, monitor their execution, and communicate. Initially, the unmanned forces will probably require cognitive support from humans in the loop. However, even rudimentary unmanned force capabilities will require a minimal ability to perceive the environment, react to commands, and plan simple actions.

Based on available information about terrain, weather, logistics and enemy forces, they must be able to plan routes satisfying practical requirements for timeliness and stealthy operation. The information available to the unmanned forces should be consistent with that possessed by the human crews: the unmanned forces should not be privy to better terrain information than would be available to a human crew member, for example.

The unmanned forces must be capable of carrying out their plans and recognizing factors that influence their outcome. They should have the capability to replan to take advantage of new information or opportunities. They must be able to detect enemy and friendly vehicles and recognize the difference. They must be able to differentiate among various vehicle and threat types. They must be able to select aim points and appropriate weapons and the must be able to lay the weapons on the targets. They must be able to evade and hide when the situation warrants it. They may need an ability to work interactively to solve tactical problems.

Since a well-integrated unmanned force component will require the mimicking of human organizational, planning, and perceptual capabilities, significant advances in the artificial intelligence (AI) state-of-the-art will be necessary for a truly seamless simulation. Presumably these capabilities will be developed in an evolutionary fashion over a number of years. In order to provide the means to evolve to a truly seamless simulation, the requirements they impose on databases and data interchange mechanisms must be anticipated and planned for now.

3 Information Requirements

The key information requirements are motivated by the need to provide simulation data to the forces consistent with that which would be used by their human counterparts for planning and operations. Standards will need to be devised for processes which transform current data formats into a form suitable for automated planning and and perception of the environment. The primary categories of information include:

- Terrain and environmental information;
- Background information for operations including friendly and enemy doctrine, tables of organization and equipment (TO&E), equipment capabilities and "signatures:"
- Information about the local battlefield;
- "Scene" data which would ordinarily be perceived by a human crew member; and
- Intercommunication between manned and unmanned forces.

Much of this data must be supplied by the simulation system. Other data may be developed by the unmanned forces during their operations and must be integrated into an internal, vehicle-centered model of the tactical environment. Each of these information categories imposes its own requirements, which we shall now consider.

3.1 Terrain Data

Terrain and environmental influences are key factors in processes for determining routes to objectives, river crossings, overwatch points, fields-of-fire. firing positions, rally points and hiding places. Current terrain databases provide relatively low-resolution data, compared to that needed for planning for individual vehicles. It is necessary that the terrain DB standard include interpolation procedures for inferring terrain characteristics at the resolution necessary for supporting the planning processes using data in standardized terrain DBs. There should be a specification for the types of interpolation that are supported. The obvious interpretation process is geometric: it must

be possible to estimate the three-dimensional "texture" of the terrain. Other types of interpolation will be necessary for reasoning about soil bases, vegetation, and so forth, that will be necessary for both planning the activities of unmanned forces and ultimately for carrying them out.

3.2 Background Information

In order both for the unmanned forces to plan competently and to enable human commanders to anticipate the activities of their unmanned subordinates, a wide variety of background information must be provided. There should be standards for the format and content of such information. These types of data will typically be normative and stable over long periods and should be provided to all players in standardized formats.

A great deal of this information will be descriptive; it will describe current enemy force organization, the equipment comprising the enemy force structure, the capabilities of both enemy and friendly equipment, and typical observables associated with equipment and activities.

Other background knowledge will be procedural; it will define typical sequences of primitive operations comprising tactics, maintenance and repair (for example, recovering a tank from a ditch), and tactical operations. The background knowledge base should also include procedural descriptions for the planning and interpretation processes themselves. Specifications for background procedures should be determined from specifications for the activities themselves. For example, if we could set a standard for real-time planning requirements, logistics effects, mobility, and terrain effects, then human planners could issue orders confidently and a multiplicity of developers could contribute to the overall simulation.

3.3 Battlefield Information

Successful tactical commanders maintain an internal model of the identities, locations, and current activities of all relevant entities in their local area of interest. This information may be acquired through data links to other friendly units or as a result of reconnaisance, routine detection, or combat. The commander also understands the volatility of such information and takes it into account in determining his tactical plans. These data are specific to the unit and represent the cognitive state of the human counterparts to the

unmanned forces. The cognitive state will result from sensing operations and direct data links, and therefore, are somewhat less dependent on the simulation databases. Even so, there should be standards for the representations of such data, particularly for representing imprecise data or information of dubious certainty. Such representations should include processes for estimating the certainty of the data as time passes: the data should probably be organized geographically.

3.4 Scene Data

In simulation programs such as SIMNET, the majority of battlefield information is perceived and interpreted by the human crew members. It is their responsibility to recognize the enemy tank in defilade and to take appropriate action. An equivalent capability will be needed by the unmanned forces. However, we do not wish to have to solve the machine vision problem in order to allow the unmanned forces to monitor and interpret their environment. Instead, there will need to be standards for a "symbolic" scene model and for symbolic presentation of information to the unmanned forces similar to standards for rendering scenes for human crews. The symbolic output will ideally be determined from the object database used to generate the synthesized renderings used presented to the humans. Additional routines must be provided to determine what is actually perceived from this data. For example, the "presentation manager" for unmanned forces may require probability tables to determine whether a partially hidden enemy tank in the field-of-view of the unmanned element is actually perceived and recognized. There will need to be standards for environmental effects on such perceptions, as well as standards for the perceptual degradation caused by battle damage to sensors.

3.5 Intercommunication Standards

Today's tactical and operational doctrine emphasizes cooperation and interaction among friendly force elements, in order to create effective, local force advantages. A realistic simulation must include this intercommunication. Today, a great deal of communication involves speech and natural language. There must be a capability to handle speech, even if it means using human operators to interpret commands from human commanders and

translate them into a standardized format for electronic transmission to the subordinate unmanned forces. Such communication protocols should include typical kinds of human speech interactions, such as the recipient requesting additional information. We do not wish to solve the computer speech problem here, either, therefore there must be standards for communication that provide representations for the typical information exchanged in battlefield communications.

4 Summary

This short position paper has outlined a wide-ranging set of information requirements aimed at making the integration of manned and unmanned forces as seamless as possible. This is an extremely ambitious objective, and in many cases, it is impossible to define standards today; we do not have the requisite understanding of the true requirements for unmanned force capabilities. However, we can begin to outline requirements and standards for terrain data and for information comprising the scene presented to the unmanned forces. These standards should cover data formats, database contents, and transformation processes necessary for reformulating simulation data for presentation to the unmanned force elements. Since we can anticipate the necessary automatic planning and sensing capabilities to gradually evolve and improve over time, a key requirement of any standard is that it be extensible.

Large-scale, real-time simulation programs such as SIMNET, allow a large number of players to share the same "game board," and offer an unprecedented potential for inexpensive training, doctrine definition, requirements specification, and research into automation. Standards defined for such systems must take this rich potential into account and must enhance the ability to realize these potential benefits, while providing a framework which enables multiple vendors to contribute effectively to the overall mission. The standards must provide a stable, predictable environment, in order to enhance the utility and effectiveness of the simulation.

As we have noted, many of the desirable automation capabilities are beyond present day state-of-the-art. However, programs such as SIMNET can provide an extremely valuable testbed for the development of these capabilities, and current standards must take these future developments into account.

Database Requirements for Semi-Automated Forces in SIMNET

David Payton, David Keirsey, David Tseng
Hughes Research Laboratories
Malibu, CA

The database requirements for Semi-Automated Forces (SAFOR) vehicles in simulation environments such as SIMNET will depend primarily upon the realism that is required of SAFOR behavior. We have therefore defined two basic classes of SAFOR operation, each with it's own distinct database requirements:

1) Group operation
2) Individual operation

These two classes of operation are intended to account for the fact that complete detail of each SAFOR vehicle is neither required nor desired. Nevertheless, there are still occasions in which simulation of each individual SAFOR vehicle is the only way to provide the kind of realism that will be able to convince a human participant that he is playing against another intelligent opponent. In the following discussion, we describe each mode of operation and discuss their corresponding database requirements.

Group operation occurs whenever a SAFOR vehicle is not in "contact" with another vehicle. The definition of the word "contact" is discussed later. Group operation implies that groups of vehicles and not individual vehicles are to be simulated. At this level, SAFOR units are controlled at the abstraction of a Platoon, Company, Battalion, or Brigade. The reasoning for this level involves determining formations, planning routes, and coordinating between groups. The terrain database must therefore entail all the standard features of a DMA database, including elevation, ground cover, and drainage. A grid-based representation is also probably appropriate at this level.

Individual operation occurs whenever a SAFOR vehicle is in "contact" with another vehicle or is a member of a Platoon for which any member is in contact. At this level, we have to simulate:

1) Perception of the environment
2) Reasoning and decision-making of the crew
3) Dynamics of the vehicle
4) Control of the vehicle
5) Network communication

One constraint that we cannot seem to escape is that each individual SAFOR must be able to perform most of the same functions that a manned unit performs. All of the environmental features which are to be observable to the manned units must also be available to the SAFOR units if they are to behave as if they were manned. Similarly, the SAFOR units must appear to others as if they are the same as manned vehicles. The only part that is different is the human-machine interface. The database required for individual SAFOR operation must therefore be at least as detailed as the database used in individual manned simulators.

CONTACT:

The definition of contact defines when particular SAFOR units should be simulated as individuals and when they may be simulated as groups. To be in contact is to be sufficiently close to a manned unit that detailed individual simulation is warranted. The following constraints must be considered in determining whether or not a SAFOR unit is in contact:

1) A SAFOR should already be simulated as an individual before any manned unit can see it.
   a) This is definitely true if the manned unit is a crew of trainees
   b) We may also want this to be true for stealth-flying-carpets
2) All four tanks in a platoon may need to be considered in contact if any one of them is. This would seem appropriate since they need to work together as a team.
3) A completely disabled SAFOR vehicle need not be considered in contact

Based on the above constraints, it seems that whenever a manned unit comes within a certain radius of a SAFOR platoon, all the vehicles in that platoon should become individual SAFORs.

The greatest database demands will arise from simulating perceptual inputs for individual SAFOR units. In order for a SAFOR to truly behave as if it were a manned unit, it must receive all the perceptual inputs that are deemed sufficiently relevant to be provided to the manned units. Perceptual input to SAFOR units must therefore be similar to the perceptual input to a manned unit except that the SAFOR needs the input in a significantly different representation. Rather than just graphics on a screen, the SAFOR needs a representation of what objects it sees. This can vary in complexity depending upon how much about an object's parts we wish to make available, and how accurately we wish to simulate the effects of partial occlusion of objects. The fidelity of the SAFOR's reasoning processes will depend directly on degree of realism provided by its perceptual input. For example, if a SAFOR is not allowed to recognize whether or not it sees the front or back of a partially obscured tank, this may limit it's ability to reason about such a situation. The following is a list of the basic perception requirements in order of priority:

1) Object-based representations of the following:
   - Other vehicles within direct line of sight
   - Man-made objects within direct line of sight
   - Obstacles within direct line of sight
   - Visible signs of explosions and fire
2) Sounds -- nearby explosions, other vehicles
3) Terrain features such as hills, valleys, gullies, etc. (limited to direct line of sight) (this may be needed for hiding and other tactical maneuvers)
   Also, vehicle pitch and roll data.
4) Parts of objects, such as the direction and motion of a turret on another vehicle
5) Clouds, smoke, chaff, and other features which obscure visibility.
   - Their effect on visibility to other features in the environment must be modeled.
   - They are also important cues about what's going on in the environment.

These requirements are needed in addition to the standard terrain representations that are currently used for modeling vehicle dynamics in manned SIMNET nodes.

The database must be designed so that intervisibility between objects can easily be computed. Unlike manned SIMNET nodes which use a graphics engine to determine the visible scene, the SAFOR units will require an object-based description of the visual scene. Thus, rather than first creating a visual scene and then extracting the appropriate description, we will want to generate the symbolic description of the scene directly. Most likely, this will result in a vehicle-centered top-down view of all objects within line-of-sight to each SAFOR vehicle. We will therefore need not only to compute intervisibility from a SAFOR unit and all terrain surfaces, we will also need to compute intervisibility to any

static and moving objects. Features such as smoke and clouds which partially obscure objects may complicate this process, requiring that some probability of object detection be included in their representation. Also, partial occlusion of objects may require that object parts be represented to some degree.

U. S. Army Engineer Topographic Laboratories
Research Institute
Autonomous Technologies Division
Fort Belvoir, Virginia 22060

COORDINATE SYSTEM CONVERSIONS APPROXIMATE METHOD

Purpose. This paper describes an approximate method for converting coordinates given in one ground coordinate system into values pertaining to a different ground system. The method is simple, fast and generally accurate for most applications. The discussion given is limited to conversions between geographic, Universal Transverse Mercator (UTM) and local rectangular (cartesian) coordinate systems.

Background. A "database" user may frequently encounter the case where the information stored in the database is given, for example, in a geographic coordinate system (latitude, longitude and height) whereas the desired data format might be UTM (Easting, Northing and height). A typical case is that of the standard DMA DTED database where an elevation value of the terrain is given at some incremental spacing in latitude and longitude. In such a case, a means must be available for converting coordinates from one system to another.

Another important case where coordinate conversions are required is when different systems, such as simulators, are connected together via networks to share in some sort of exercise such as battlefield simulation. In this case, there is a good chance that position, range, azimuth, etc. is reported as input to the exercise by weapon systems, sensors, other simulators, etc. in various coordinate reference systems.

The coordinates of the database elevation values and the inputs to the battlefield simulation exercise, as examples, can be converted to other coordinate systems using rigorous mathematical computations. This is a time consuming process, however, that involves hundreds of floating point operations and several hundred lines of code. The accuracy requirements may not justify the expense of the rigorous computations.

The geographic system is the common denominator between the various coordinate systems. UTM's are derived from geographics by projecting geographic points on the "round" Earth onto a cylinder that is nearly tangent to the Earth and has it's axis in the plane

of the equator. The local rectangular values are obtained by converting spherical coordinates, and heights above the spheroid, to cartesian coordinates (x,y,z) relative to a plane surface that is tangent (or secant) to the earth at some selected, local geographic origin.

Since there is a specific mathematical relationship between the various coordinate systems, it is possible to approximate the rigorous relationship with simple, but adequate, polynomials that can be used to perform the desired coordinate conversions. The advantage of the approximate approach is that it is accurate, fast and can be implemented with a minimum of code. The following polynomials, as examples, can be used for the purpose:

$$xc = a_0 + a_1*xm + a_2*ym + a_3*xm^2 + a_4*ym^2 + a_5*xm*ym$$
$$yc = b_0 + b_1*xm + b_2*ym + b_3*xm^2 + b_4*ym^2 + b_5*xm*ym$$

$$xc = a_0 + a_1*xm + a_2*ym + a_3*xm^2 + a_4*ym^2 + a_5*xm*ym + a_6*xm*ym^2 + a_7*ym*xm^2 + a_8*xm^2*ym^2$$
$$yc = b_0 + b_1*xm + b_2*ym + b_3*xm^2 + b_4*ym^2 + b_5*xm*ym + b_6*xm*ym^2 + b_7*ym*xm^2 + b_8*xm^2*ym^2$$

In the above equations the xc and yc values are the changes to the coordinates in the desired system that correspond to the changes in values (xm,ym) in the given coordinate system. The changes are measured relative to some arbitrary origin within the physical area of the database. The "a" and "b" coefficients are those that must be determined in order to use the polynomials to convert xm and ym to corresponding xc and yc values. A method for computing the coefficients and using the polynomials is discussed in the following section.

Method. In order to determine the coefficients of the transformation polynomials, it is first necessary to compute precise desired coordinate changes (xc,yc) for a set of given coordinate changes (xm,ym). For example, if geographic coordinates are to be converted to UTM's, an array of nine (3x3) or more geographic values can be selected which outline, or are greater than, the database area in question. Using the rigorous conversion formulas, the absolute geographic values can be converted to precise UTM values.

One of the geographic points is selected as the origin of the polynomial conversion. The origin should be centered approximately within the data base area in order to minimize the magnitude of the changes relative to the origin. The computed UTM values for the geographic origin will be the origin of the UTM system. Next, the coordinates of the geographic origin are subtracted from all of the fabricated geographic values and the differences converted to units of seconds. Likewise, the coordinates of the UTM origin are subtracted from all of the UTM values. With these two sets of data, a least squares technique is

used to compute the coefficients of the above transformation polynomials.

The coefficients and polynomials can now be used to convert other geographic coordinates to equivalent UTM values. Each geographic coordinate must first be subtracted by the geographic origin used in the fabricated data and the differences converted to seconds. The right side of the polynomials can then be evaluated to determine the corresponding differences from the UTM origin (xc,yc). These differences are added to the UTM origin to obtain absolute UTM values for the geographic point in question.

Transformations involving geographic-to-UTM, and the inverse, do not involve the height of the point above the spheroid. Transformations involving local coordinates, however, must consider the height of the point. Therefore, the above transformation polynomials must be changed accordingly. The change required involves a multiplication of the xm and ym terms in the polynomials by a "dz" factor. If local rectangular coordinates are to be converted to geographic or UTM values, dz is defined as:

$$dz = 1.0 - (h/R)$$

where:

h .... height of point above sea level
R .... Radius of the earth

A local rectangular zm coordinate can be converted approximately to a height above sea level by adjusting it for earth curvature according to:

$$h = zm + (xm^2 + ym^2)/2R$$

If geographic or UTM's are to be converted to local values, the dz term is defined as:

$$dz = 1.0 / (1.0- h/R)$$

Now the xm and ym coordinate changes on the right side of the polynomials are:

$$xm = xm'*dz$$
$$ym = ym'*dz$$

where: xm',ym' .... the true UTM, geographic or local changes.
xm, ym ....the changes modified for height above sea level.

The dz term has the effect of reducing the local x and y coordinates for a point above the spheroid to corresponding values for a point directly on the spheroid. This eliminates the affect of the height variable from the coordinate transformations.

Accuracy. The accuracy of the approximate polynomials is dependent primarily on the size of the geographic area over which they are applied. The systematic relationship between coordinate systems becomes more complex as the physical area increases, and there is a point where the polynomials discussed above may become too inaccurate for the purpose.

In order to judge the accuracy that can be expected by using the polynomials, a set of 25 geographic coordinates were fabricated to cover areas of 15, 30, 45 and 60 minute quadrangles on the terrain surface. The origin was taken at the center of each quadrangle. Precise UTM values were determined for the 25 geographic points using the rigorous conversion methods and the "a" and "b" parameters were evaluated for both the 6- and 9-term polynomials. The maximum errors obtained in "fitting" geographic changes to UTM changes were as follows:

	AREA OF COVERAGE (MIN)							
	15 (27x22km)		30 (55x44km)		45 (82x67km)		60 (110x89km)	
	max error (meters) in east and north							
	e	n	e	n	e	n	e	n
6-term	.014	.006	.119	.049	.375	.166	.889	.395
9-term	.001	.000	.007	.001	.025	.004	.059	.010

The accuracy results show that for an area as large as 60x60 minutes of latitude and longitude, both the 6- and 9-term polynomials can be used to transform geographics to UTM's with errors not exceeding 1 meter per axis. These results are typical and apply to geographic-to-local, local-to-UTM and their inverses as well.

Speed. The computational speed of the polynomial method, as compared to the rigorous method, was evaluated by performing a geographic to UTM transformation for 1 million coordinates using both methods. The computations were performed on a Silicon Graphics 4D/80GT computer and a Vax 780. The results are as follows:

	GEOGRAPHIC TO UTM CONVERSION (1 MILLION PTS)	
	SG-4D/80GT	VAX 780
6-term	13"	141"
9-term	18"	218"
RIGOROUS	90"	1461"

The 6-term polynomial can be executed about 7 to 10 times faster than the rigorous method and the 9-term polynomial is about 5 to 7 times faster, depending on the computer used.

621-61-90

VICTORY
INTEGRATED SYSTEMS, INC.

POSITION PAPER

SENSOR AND COUNTERMEASURE MODELING IN DISTRIBUTED SIMULATION

MEMORANDUM
M-90-01

Submitted to:

NETWORK STANDARDS WORKING GROUP

Submitted by:

VICTORY INTEGRATED SYSTEMS, INC.
6120 Paseo Del Norte, Suite O-2
Carlsbad, CA 92008

by
Peter Thompson

January 1990

6120 Paseo Del Norte, Suite L-2 • Carlsbad, California 92009
Post Office Box 1547 • Carlsbad, California 92008
FAX (619) 431-9668   (619) 431-9667

1.0 INTRODUCTION

The training and simulation community is placing increasing emphasis on high fidelity representations of sensors and systems, both friendly and threat, to provide design, tactics development, and training support for current advanced and future weapons systems. The emphasis covers simulation objectives to achieve effective team training, mission rehearsal, and rapid tactics development and validation tools. Objectives, particularly in the areas of mission rehearsal and tactics development, stress across-the-board modeling fidelity, especially in the areas of sensor modeling and dynamic observables generation. Many of these needs are identified in Reference 1. High fidelity dynamic sensor modeling is critical to adequate representation of system performance in hostile countermeasure and Directed energy Weapon (DEW) environments which must be anticipated. These models must be sensitive to a broad range of operating conditions to achieve objectives for mission rehearsal as well as effective training of crew for complex battlefield environments.

VICTORY INTEGRATED SYSTEMS, INCORPORATED, is a small business in California with significant background in weapons systems engineering and simulation, and aircrew training. Our background in avionics design and simulation, as well as training and simulation for training, provides a solid base for commentary and recommendations for enhancements to distributed simulation network protocols and requirements. This paper addresses a broad set of needs for higher fidelity sensor and sensor countermeasures modeling to support emerging advanced capabilities in the combined forces simulation arena.

2.0 PROBLEM IDENTIFICATION

One of the significant problems facing the SIMNET program and related systems such as AIRNET, as alluded to above and in Reference 1, is the representation of sensor performance at sufficiently high levels of fidelity to satisfy a range of man-in-the-loop (MIL) simulation objectives which includes position training, weapons system training, team training, mission rehearsal, and tactics development. While many objectives can be satisfied by the simplified treatment of sensor performance which is currently embodied in the SIMNET program (i.e., detection probability versus range functions driven by input data tables), there are a number of issues which can not be addressed with models of this level of fidelity. Perhaps foremost among these are objectives which relate to the interaction of multi-spectral sensor suites with countermeasures of various sorts. These objectives include training and tactics development for ECM environments (e.g., ECM and ECCM employment doctrines); employment of non-cooperative target recognition (NCTR) systems, including RF-based, passive sensor/signal processing, and imaging sensor approaches; and operations in directed energy weapon (DEW) environments.

Simulation communications protocols which support adequate models for these simulation objectives must be developed with a careful eye not only to the ultimate fidelity of the simulation models, but also with consideration of the communications band width limitations which are inherent to distributed simulation. The implications of each of these objective domains (ECM/ECCM, NCTR, and DEW) for distributed simulation, contrasted in particular to current SIMNET protocol approaches, may be expanded as follows.

ECM/ECCM (and IRCM) interactions with sensor suites are both highly dynamic and highly sensor/target dependent. Available techniques include (in the most general sense) barrage noise jamming, deception jamming, expendables (including expendable jammers), and IR modulators. These may be employed with signature reduction techniques (e.g., plume signature reduction through injection of chemicals into the plume) and reduced signature vehicles. Effective application of ECM techniques may require coordinated maneuver or other reaction (e.g., throttle down, turn away from observing threat line of sight, etc.). The timing for initiation of an ECM technique is also critical to its success. Likewise, the advent of multi-spectral sensor suites across the battlefield and the intrinsic ECCM functions of current sensors demand attention in simulation. Operator recognition of ECM employment and the proper ECCM response can be critical to mission performance. These responses may be as simple as a mode change (perhaps even automatic), or may require intensive manual intervention (as in the case of manual tracking with an optical or electro-optical device to overcome track loss with a primary radar tracking sensor). The proliferation of ECM/ECCM and IRCM options and combinations makes a detection probability versus range table approach infeasible. Data base growth to accommodate not only sensor/target pairs, but ECM/ECCM/IRCM tactics combinations, will rapidly exceed reasonable capacities both for storage and configuration control of simulation data. Adequate representation of ECM/ECCM/IRCM in distributed simulation networks demands enhanced network protocols for communication of target signature and/or target/ownship state, to support higher levels of fidelity in sensor modeling.

NCTR systems are typically of two types: signal processing based or image based. Multi-sensor integration may also be used to enhance general target recognition capability. All NCTR systems are highly dependent on sensor/target geometry and target state. As with ECM/ECCM, proliferation of data bases to represent sensor performance in these areas may be impractical.

DEW environments offer some unique challenges for distributed simulation. Foremost among these is the nature of DEW impact, which can not typically be represented by simple kill categorizations (M-kill, F-kill, or C-kill). Instead, DEW impacts are typically more subtle, frequently manifesting through degradation of sensor performance. This is especially true of laser threats. Such degradations may be either temporary (present only for the duration of exposure) or permanent. These degradations are particularly important for the presentation of images to a simulator crew. Representation of reasonable degraded images intrinsically levies requirements for reasonable representation of imaging sensor performance, and this requires signature representation enhancements and on-line sensor models. Degraded image representation can not be adequately supported with simple range/performance curves. DEW effects are highly dependent on access of the weapon to the sensor. Among other things, this access is driven by sensor line-of-sight and sensor/threat geometries, and is therefore highly dynamic. DEW simulation protocols, as well as sensor observables protocols, must be developed for representation of subtle DEW effects in a distributed simulation environment.

All of these issues are critical for the development of enhanced sensor/observable models in distributed simulation to achieve simulation objectives for advanced systems and

advanced tactics both on the surface and in the air. High dynamics of signature and target accessibility imply not only additions to simulation protocols for additional data for models, but also enhancements to the basic approach for player propagation and state information communication which have been defined for SIMNET. The current SIMNET approach, which embeds player state projection models in each simulation station, is an important design which limits communications traffic effectively, but levels of fidelity in physical state preservation and calibration across the distributed simulation network must be carefully reviewed. It is VICTORY's belief that enhancements will be required for higher fidelity battlefield simulation, especially as more and more aircraft are integrated into the simulated battle. These issues are already under consideration for SIMNET.

Underlying even these issues is the availability of basic signature data to support enhanced sensor/observable representation. Project 2851 is chartered with the development of a Standard DoD Simulator Data Base/Common Transformation Program. Therefore, Project 2851 should ultimately provide the source of all terrain and target signature data. Unfortunately, the current 2851 scope does not provide for sufficient multi-spectral signature data. Signatures are required, at a minimum, for visual, two near-IR bands (for GEN II and GEN III night vision equipment), mid-IR, far-IR, at least two RF bands, and two microwave bands. Additional signature issues may relate to optical cross-sections under active illumination. Current 2851 designs provide only for the visual band, a single IR band, and a single RF band. Project 2851 must be more responsive to future needs for multi-spectral sensor simulation. The data base needs encompass not only environmental and terrain data bases, but target signature data bases as well.

Thus, three main problem areas for distributed simulation network protocols are identified:

1) Sensor/observable communications protocols must be enhanced.

2) ECM/ECCM action/response protocols must be developed. This area includes as well IRCM and DEW simulation action/response protocols. These protocols must ultimately address not only signal/energy distributions, but vulnerability/target response information and signature modification due to exposure as well.

3) Underlying data bases under development via Project 2851 must be reviewed and enhanced for future requirements in multi-spectral, high fidelity sensor simulation.

3.0 SIMULATION AND PROTOCOL REQUIREMENTS

This section outlines a set of protocol requirements which support the simulation needs discussed in the previous section. The requirements for protocol update fall into four categories: Sensor Support Protocol Data Units (PDU), CM Employment PDU's, Emissions PDU's, and DEW Employment PDU's. The overall strategy for enhanced network communications in this area is illustrated by Figure 1. As illustrated, Sensor Support PDU traffic is initiated by an Information Request Message. This message will be sent to potential sensor targets as determined by simple sensor coverage region checks. In return,

the "observed" simulator responds with a series of messages which provide necessary information based on the sensor types being used by the observer. This enables traffic to be limited to the sensor/target pairs and specific data types required. In addition, the observed simulator will issue CM employment messages and emissions messages on an as-required basis which support enhanced sensor modeling and interactions. Also issued by the "observed" simulator are DEW Employment messages. These messages are separable from the other types, since they are not related to the information request message, which indicates that a particular target is being tested for sensor detection, but are simply related to offensive targeting potential for the DEW simulator. These messages are discussed here, however, since they have implications for sensor modeling via DEW effects simulation.

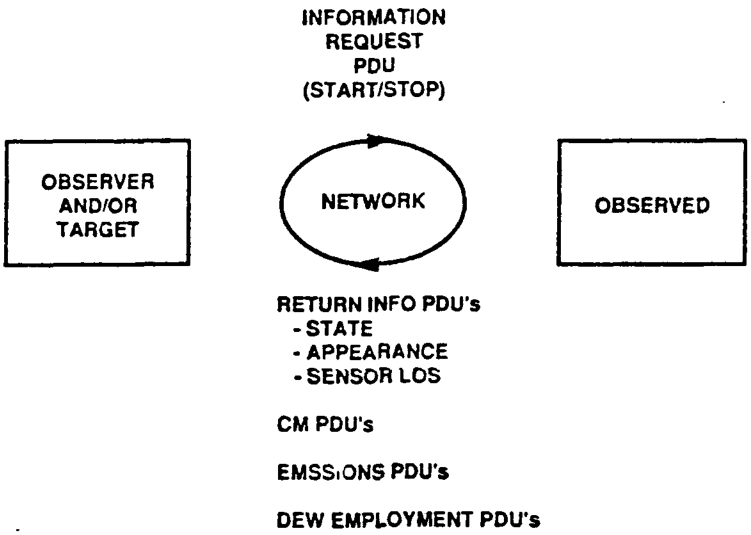


Figure 1. Network Protocols for Enhanced Sensor, Countermeasure, and DEW Simulation

One preferred simulation architecture for implementation of advanced sensor models is the incorporation of a Sensor and Emitter Interface System (SEIS) in each simulator on the network. The SEIS is a bundled simulation module incorporated in individual simulators to perform sensor and emitter performance and effectsw modeling. The SEIS handles communication traffic and provides the sensor simulation for each equipped simulator. The SEIS communicates with the "outside" simulation world via the network, and passes observables characterizations on to display and other simulator elements on the "inside". Observables characterizations include image data as well as non-imaging sensor data. The SEIS also incorporates explicit representation of CM effects and DEW effects as required for simulation objectives. Different SEIS designs are required for MIL versus Semi-

Automated Forces (SAF) simulators, due to the different observables modeling needs for these simulators. For example, SAF simulation does not requrie explicit visual representation of external scenes, but does require representation of visual performance. It is important, however, that the external communications interface (i.e., the network protocol) be consistent for all forces (both MIL and SAF). Internal interfaces should also be standardized to the greatest possible degree in order to minimize the effort in tailoring internal SEIS functions to the desired levels of fidelity for each simulator on the network.

Recommended content of each of the above delineated message types is discussed in the following subsections.

3.1 SENSOR SUPPORT PDU

Sensor support protocol enhancements revolve around improved vehicle dynamics and expanded dynamic signature representation. The concept for sensor support is that observer simulators issue information request messages to all simulators which are candidates for observation. This candidacy can be checked simply by sensor coverage region definitions, much as is currently defined for sending Vehicle Appearance PDU's in SIMNET. The information request message must identify the observing simulator and the types of sensors which may be used for observations.

In response to this message, the observed simulator returns the following:

- Enhanced appearance PDU's. These must include enhanced state information. The state information must include not only position and velocity, but also acceleration, and for aircraft must include orientation (roll, pitch, and yaw (RPY) and RPY rates). These PDU's support higher order projection models which will improve dynamic performance in the distributed simulation and support more detailed sensor performance considerations for such things as pulse-doppler radar, low signature vehicles, and NCTR sensors/modes. Enhanced appearance PDU's must also include not only throttle setting, but current vehicle temperature (to support IR sensor modeling).

- Observed Sensor State PDU's. In the case of active optical sensor operation, the observed target's sensor state, especially for any EO or IR sensors, may be critical to determination of signature (i.e., optical cross section). This state includes description of sensor types, line-of-sight (LOS), LOS rate, and scan pattern or scan mode definitions. Depending on the level of fidelity required in predicting sensor (pointing) states of other simulation players, specific sensor LOS prediction/projection models will be required (analogous to those employed for position projection in SIMNET).

These basic message sets can support high fidelity sensor modeling at the sensor simulator. The sensor simulator must carry on-line static signature data bases for each potential target type. These data bases may include spectral signatures in a variety of sensor pass bands, including .4-.7 μm (visual), .4-.9 μm (GEN II Night Vision Equipment), .55-.95 μm (GEN

III Night Vision Equipment), 1-3 μm (typical missile seeker), 3-5 μm (mid-IR imager or IRST), and 8-12 μm (far-IR imager), as well as any required radar cross section data. Signature data may include emissivity, radiance, and reflectance data. Plume and hot parts signatures as a function of throttle setting should be included. Coupled with appropriate levels of detail in environmental models and data bases, virtually any desired level of fidelity in sensor modeling can be supported with protocol enhancements as described above.

3.2 CM EMPLOYMENT PDU

CM employment PDU simply inform an observing sensor that a CM has been employed, and identify the CM type and mode explicitly. Models and data bases resident at the observer's simulator incorporate the explicit effects of the CM on sensor and system operation. A CM employment "stop" message is also required. Since many CMs (i.e., all forms of expendable CM) become simulation players with their own independent dynamics, the CM employment PDU(s) must make provision for the initiation of new players to represent the expendable CM.

3.3 EMISSIONS PDU

The emissions PDU are separated from Sensor Support and CM employment PDU largely because they are not related to prior receipt of an information request message. Instead, they are sent when the player emitters are activated. Enhancements of the emissions PDU already defined for SIMNET may be required. In particular, extensive use of laser designators and rangefinders in the field, and the advent of laser warning receivers (LWR), require that emissions messages be defined not only for RF but also for laser emissions. This should be a straightforward enhancement of the current protocol for emissions. The emissions PDU should be forwarded to any simulator in a coverage region which is of a type which has receivers which are sensitive to the emission. This can be accomplished either by a multi-cast (controlled by the issuer of the emission PDU), or by a broadcast (filtered by the receiving simulators).

3.4 DEW EMPLOYMENT PDU

DEW employment PDU must identify the targeted simulator(s) and the parameters of the DEW beam at the target. These parameters must include the spatial and temporal distribution of beam intensity at the target, incorporating such effects as beam wander (due to pointing and tracking system inaccuracies and aimpoint selection).

4.0 RECOMMENDED ACTIONS

Actions to be taken by the Network Standards Working Group are clearly recommended by the discussion above. First, in the interest of supporting enhanced sensor models for realistic simulation, and for additional capacity to support detailed simulation objectives, modifications to the simulation protocols such as those defined in Section 2 above should be considered. Second, it is vital that feedback be provided to Project 2851 in the area of

signature data base development and enhancement. These steps can help to insure that distributed simulation systems such as SIMNET and AIRNET have the capacity to accommodate increased simulator node computational capacity and detailed simulation and training objectives.

There are several study activities which should be undertaken to refine network protocols in support of higher fidelity model requirements. These include the following.

1) Timing studies for signature dynamics should be undertaken. The nature of signature dynamics with respect to target presented aspect should be examined in depth. Additional consideration should be made of the dynamics of optical cross section and how they relate to sensor scan patterns and modes and conclusions should be reached concerning the most compact method of communicating required information on the network.

2) Fidelity of CM response models is critical to effective training in the use of CMs and CCMs and the rehearsal of CM-related missions. Historically employed CM models range from greatly oversimplified (e.g., P_K draw-down factors) to complex, pulse-by-pulse signal processing emulations. A middle ground of CM characterization is required and should be identified and specified.

3) SEIS interfaces to Computer Image Generation (CIG) systems are a critical area. These interfaces must support not only visual image generation, but imaging sensor output image generation in general. The means of data transfer must be specified, and must support not only basic sensor performance, but also degraded sensor performance, especially in CM and DEW environments.

4) Target data base development is an important element of a successful effort to upgrade sensor modeling fidelity. Target vulnerability data base development is critical to DEW simulation. The needs for target data bases for visual presentation (which has been emphasized to date), and the needs for other signature and vulnerability data bases must be carefully reviewed for consistency. For instance, if current target signature data bases do not support body occultation of radar glint points, and these glint points are required, an additional glint point occultation data base must be developed. Physical consistency of this data base with the existing data base must be maintained. In the long run, a fully self-consistent and self-contained target data base must be specified and developed.

VICTORY has taken steps towards the development of many of the specifications and much of the information called for in this paper in various efforts, including support for the Special Operations Forces Aircrew Training System (SOF ATS, see Reference 2). We anticipate an increasing involvement in the important area of distributed simulation.

5.0 REFERENCES

1. "Naval Aviation TACAIR/ASW Aircrew Simulator Training Requirements Analysis (Unclassified Version)", CNO Memo: Ser 593D/9U598436, 12 June 1989.

2. "Modular Threat Environment Simulation Design Discussion", L. Jobson, VICTORY Report, 31 May 1989.

L02-01-90

VICTORY
INTEGRATED SYSTEMS, INC.

POSITION PAPER

SEMI-AUTOMATED FORCES MODELING FOR AIRCREW MISSION REHEARSAL TRAINING

MEMORANDUM
M-90-02

Submitted to:

NETWORK STANDARDS WORKING GROUP

Submitted by:

VICTORY INTEGRATED SYSTEMS, INC.
6120 Paseo Del Norte, Suite O-2
Carlsbad, CA 92008

by
Lawrence B. Jobson

January 1990

1.0 INTRODUCTION

The driving requirements for future tactical aircrew team training simulations are based on the need for mission rehearsal and rapid turn-around tactics development while on operational deployment. VICTORY INTEGRATED SYSTEMS, INC. (VICTORY) continues to be actively involved in the development of high fidelity simulation modules for aircrew training simulations. Our position paper for improved Semi-Automated Forces (SAF) modeling in SIMNET is oriented toward conducting aircrew team training at the mission rehearsal level.

2.0 PROBLEM IDENTIFICATION

To incorporate an improved SAF system into the SIMNET architecture the following issues need to be addressed:

- o Increased fidelity of the "1-versus-N" projection model
- o Increased fidelity of the SAF module
- o Increased local area network (LAN) bandwidth

Previous DoD-sponsored training deficiencies studies (eg.Reference 1) have established substantial requirements for simulation fidelity upgrades in order to perform effective mission rehearsal training. Mission rehearsal simulation for aircrew team training requires a network of full-up cockpit simulation stations, reconfigurable cockpit simulation stations and high-fidelity computer-controlled (surrogate player/instructor) stations. A given air vehicle simulation that is manned must be responsive to between 10 and 20 highly responsive targets and threats that are external to the ownship and between 50 and 100 tracks the are modeled as background tracks (minimally responsive). To accommodate the degree of responsiveness required between manned and computer-controlled tracks, substantial increases in fidelity and update rate needs to be designed into the SAF system. Additionally, substantial increases in fidelity of manned station sensors and avionics software simulations must be accommodated (see References 1&2). The interactions between manned stations, computer-controlled players (missiles, other air and ground vehicles or expendable countermeasures) and/or other manned stations may need to be updated at rates up to 30Hz. Highly dynamic aspect dependent signatures for low-observables vehicles and/or aperture dependent glint return points need to be captured by the simulation. All computer-controlled players are only highly responsive to specific external players. The instances of highly responsive interactions must be managed such that high-fidelity simulation is only performed on a local basis. Message traffic between players that are engaged in highly responsive simulation must be managed outside the main SIMNET LAN. A network manager must be provided at each manned station. This "local manager" must assess the priority for high fidelity responsive simulation between ownship and a subset of all the tracks external to the ownship but within the ownship field of regard. The "net result" of the high fidelity/high update rate interactions that are on-going between the ownship and the high priority computer-controlled adversaries can be broadcast to the SIMNET LAN in a periodic but less frequent rate (say 1 to 5 Hz).

3.0 SIMULATION AND PROTOCOL REQUIREMENTS

To keep the cost of effective mission rehearsal aircrew team training as low as possible, a substantial number of "players" must be computer controlled. Since the interaction between each manned station and the external environment must be functionally identical, what is required is a universal external world simulation module that provides for the 1-versus-N projection features of the current SIMNET structure as well as for the generation of "Virtual Player Nodes". A virtual player is a computer-controlled player that has been assigned to a given manned station for the purpose of accommodating pairwise high fidelity responsive simulation. One example of a virtual player being assigned to a manned is during the launching and guiding of air-to-air missiles. If the same missile eventually acquires the target using internal sensors, then this same missile can be assigned as a virtual player node at the target's simulation node. If the target is not a live participant, then the virtual player node will be maintained by the manned station that launched the missile. Figure 1 illustrates an example of the manner in which a manned player can "spawn" computer-controlled players. Figure 1 also highlights the fact that computer-controlled players must have adaptive fidelity that is dependent on the apparent external situation.

Figure 1. Virtual player "spawning" in tactical air vehicle trainer positions

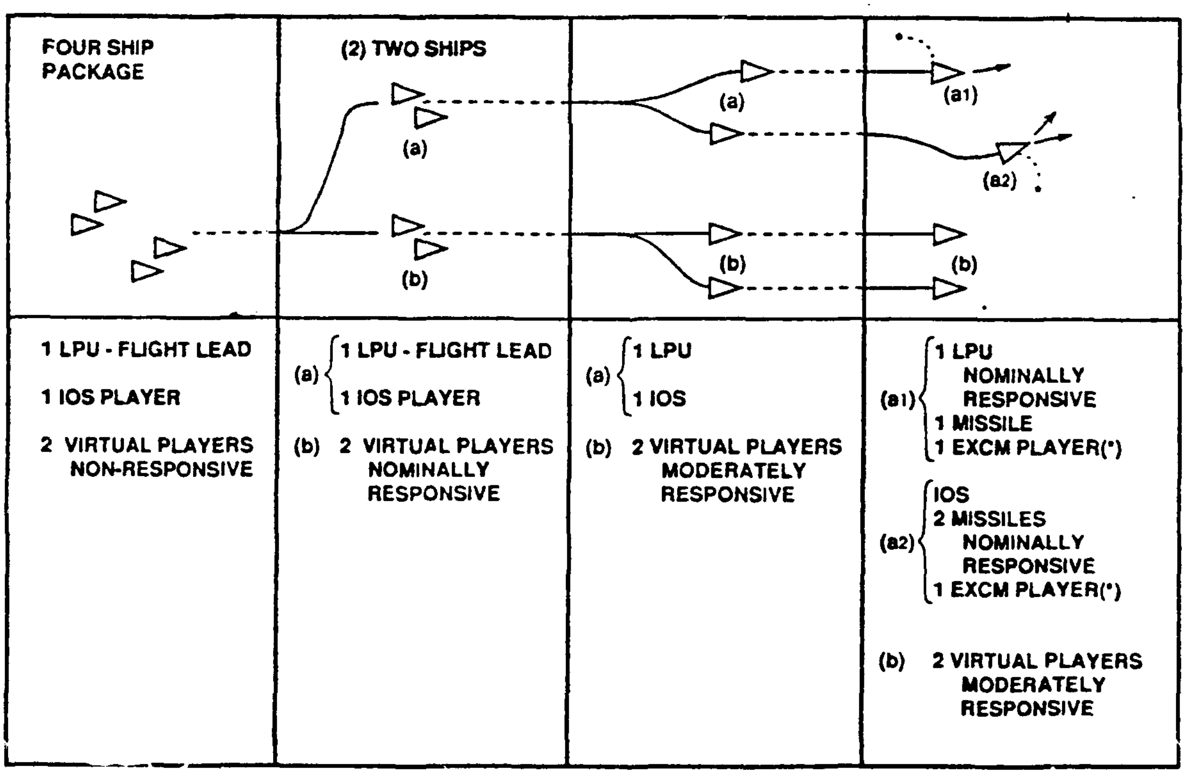


Table 1 summarizes the functional training procedures that must be accommodated within the mission rehearsal simulation. Multi-ship cooperative(data linked) and coordinated (communication channel linked) must be stressed. The primary drivers (see References 1&2) of effective aircrew mission rehearsal simulation are sensor performance, observables generation and virtual player operator response modeling. References (2&3) discuss VICTORY's position on the design of a Sensor and Emitter Interface System (SEIS) and an Operator Response Model for inclusion in the SIMNET architecture. Table 2 summarizes the requirements for the SEIS module.

Table 1. Tactical Training Procedures Required for Mission Rehearsal

Sensor & Emitter System Management
- Target/Threat Acquisition & Track
- Track State Estimate
- Signature Management
- Post Launch Missile Guidance Update
- Threat/Target Recognition & ID
- threat State Defection
- IFF Interrogation
- Multi-Sensor Cueing/Mode control
- Missile Launch Detection
- ECM Response Selection
- Multi-Sensor ECCM Control
- Cooperative Sensor Operations (Bi-statics...etc.)

Situation Assessment
- Raid Assessment
- Target/Threat Prioritization (single & 2-ship)
- Threat Response Intent Determination
- Offensive & Defensive Launch Zone Recognition
- Relative Energy Assessment
- Threat Missile Susceptibility Assessment
- ECM Effectiveness Determination
- Kill Assessment
- Semi-Automated System Prioritization Monitoring

Tactical Response Execution
- Cooperative Operation Rendezvous
- Attack Steering Path Selection
- Cued Steering Flight Path Control
- Waypoint Steering
- TF/TA & Night Navigation
- Threat Missile Launch Awareness
- Threat Launch Platform Detection Avoidance
- Threat Missile Evasion
- ECM/EXCM Response Selection
- Disengagement Maneuvering

On-Board Diagnostics Response
- System Failure Response Recognition
- Emergency Procedures Responses
- Degraded Modes Operation

The baseline SEIS design is established under the premise that advanced tactics and countermeasures training requires explicit simulation (not emulation) of the EW signal environment. The SEIS module explicitly represents sensor performance on-line as opposed to using table-look-up representations. To support our approach to sensor performance the SEIS module must generate and receive signals in all the appropriate sensor passbands. All sensor performance and signature generation must be done by each player. Traditional WST simulation approaches to sensor performance and countermeasures responses must be expanded upon to enable explicit accounting of effective signal-to-noise, signal-to-jamming noise and signal-to-background clutter. The SEIS module should support the incorporation of dynamic laser/EO countermeasure signals in visual and infrared images as well as conventional RF jamming signals. VICTORY has performed all the required sensor module fidelity assessments that support our SEIS design during previous simulation design activities.

Table 2. Baseline SEIS Functions

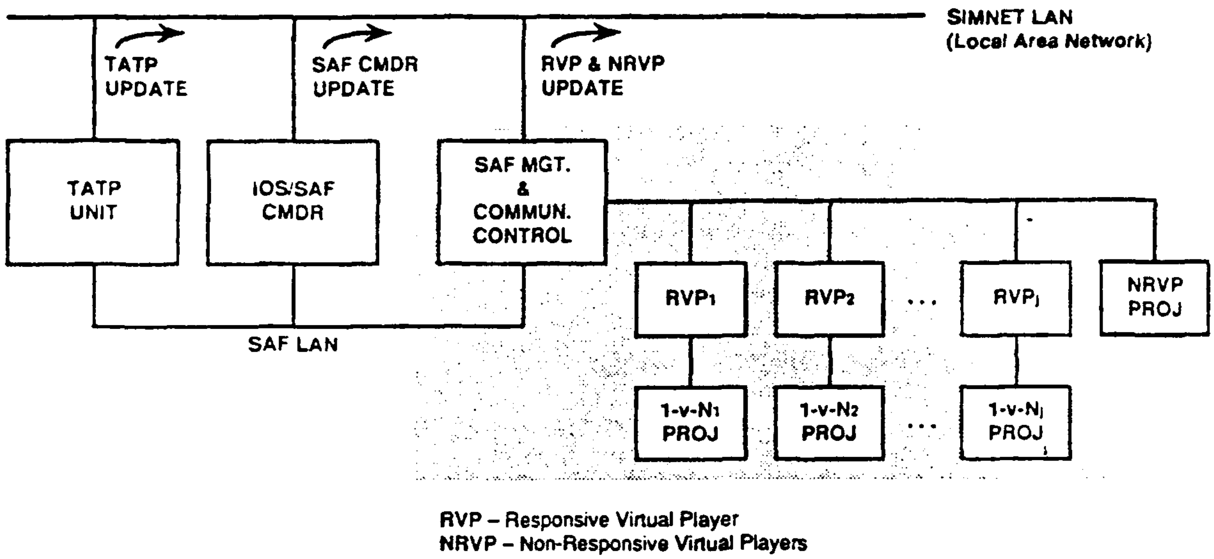
FUNCTION NAME	PROCEDURE DESCRIPTION
Signature Generation	Beam Amplitude Distribution Generation for outgoing active transmitters. Power & Divergence attributes.
Mode Control	Table driven process of frequency of transmitter and receiver channels.
Pointing Axis Control	Pointing Axis command and control for slewing and tracking sensor line of sight motion (including ESA beam formation and steering).
Signal Collection	Antenna/aperture transformation of incoming signals from passive targets, emitters (sensors and jammers) and background emissions.
Signal Transformation	Signal processing of the true target signal with respect to noise, background clutter and jamming. Thresholding to determine whether or not the true target, a false target or a track bias should be registered.
Individual Sensor Attribute	Combined effects of instantaneous signal transformation and dynamic sensor filter processing to establish the measured and derived sensor outputs. Included in this function are the expected automatic/mode controlled ECCM effects of each sensor.

Our approach to the computer-generated operator response model is based on artificial neural networks and human performance models that are maturing in the cockpit design community. Our approach is a hybrid of the value-driven expert-systems based "decision-theoretic type" formalisms and the explicit modeling of human task completion using multiple resource theory. Our overall C^3I process architecture assumes that instructor will manage "global connectivity" of the adversaries. Below the overall C3I; however, we envision a highly data driven two-tier control process that links "localized player" command and control to individual player response (human-performance-based). Allowing individual player response to be based on human performance models maximizes the availability of task procedure data bases from advanced aircraft cockpit design activities such as Cockpit Automation Technology (CAT), Army Aircrew Avionics Integration (A^3I) & Advanced Tactical Crew Station (ATCS).

The overall architecture for inclusion of the generalized SAF architecture into SIMNET is shown in Figure 2. Each manned player station (Tactical Airvehicle Trainer Position (TATP)) will have a generic SAF system that is linked to the manned player by a high capacity local area network. The generic SAF system includes an interface management module that controls the creation (assignment) of virtual player nodes, assigns fidelity level to the interaction between virtual players and the 1-versus-N projection module and

manages the periodic transmission of virtual player updates to the distributed SIMNET player network. Ideally, the hardware processor that hosts the generalized SAF system should accommodate 12 parallel channels for fully-responsive virtual player simulations. Additional low fidelity update of 50 non-responsive players should also be accommodated.

Figure 2. SAF Architecture for Tactical Aircrew Training



The generalized SAF module must be responsible for the explicit update of all virtual player nodes associated with a given manned player station. The generalized SAF module must also manage the 1-versus-N projection process. Using the concept of Fidelity Management, The virtual players and the "N" projected players all should use the same sets of performance update simulation modules.

The set of SAF performance update modules must be responsible for the following functional processes: (1)Transforming all signals that leave the ownship into observables that may be seen by external targets and threats; (2) Updating the positions, signatures and sensor pointing/mode states for all vehicles external to the ownship; (3) Generating all observables emanating from all backgrounds and external vehicles; (4) Propagating all observables to ownship location; (5) Performing all external player C^3I; (6) Computing all external player Vulnerability and Effects; and (7) Performing all Terrain Masking & Body

Occultation. Table 3 displays the functional design characteristics that we feel are critical to the classes of aircrew team training procedures listed in Table 1.

The SAF Fidelity Manager (FM) enables the SAF executive controller to dynamically adjust both update rate and module fidelity to deliver high fidelity mission rehearsal level processing at relatively low peak processing rates. Depending on the expected time rate-of-change of the output attributes associated with the SAF processes, SAF process call

Table 3. Required SAF Functions to Support Aircrew Mission Rehearsal

FUNCTION	COMPUTATIONAL PROCESS CHARACTERISTICS
Observables Generation	Vehicle Signature Generation; Background Signature Generation; Expendables Signature Generation; Line-of-Sight Intercept; Signal Transmission; Beam Amplitude Distribution
Sensor Attribute Generation	Aperture Signal Transformation; S/N; S/J; S/C; Signal Processing; State Attribute Error sampling & Filtering; ECCM Response; Body Occultation
Sensor & Emitter Position & Mode Update	Mode control; Beam Formation; Aperture Position Control; Line-of-sight Position
Aircraft State Update	Aerodynamics; Flight Control; Autopilot; Propulsion; Engine State; Armament State; Decoy State
Weapon State Update	Aerodynamics; Autopilot; Propulsion; Engine State; Fuzing State
Relative Geometry & Masking	Relative LOS Conditions; Target & Observer Aspect Angles; Terrain Mask Condition; Body & Sensor Mask Conditions
Countermeasure State Update	Mode Update; Decoy Position & Brightness; Burn Time; Chaff Dispersion & Signature
Operator Response (C^3I)	Workload Throughput; Internal SA; Decision Making; Control Initiation
Mission Avionics Software	Fire Control & Defensive Envelope State; Sensor Envelope State; Task Activity Network Recall; Tan Recognition; Flight Path Mgt; Sensor Mgt.
Multi-Sensor Fusion	Multi-Sensor State Generation; ID estimation
Vulnerability & Effects	Conventional Weapon Component Damage; System Response Network; Degraded State Determination

rates can be varied between .5 and 30 Hz. The track prioritization and fidelity management processes are based on the following:

(1) Relative Geometry (Including Time-to-Go T_{GO})
(2) Explicit Track Designation (If Any)
(3) Implicit (SA) Prioritization (If Any)
(4) Rate of Closure to "Situation Envelopes" (MLE's, Defensive MLE's, CM effects envelopes)

It should be noted that sensors "see" hundreds of emitters at the same time. These emitters are prioritized before the aircrew "sees" them; therefore; the estimate of 50 prioritized tracks represents the effective (priority) tracks that must be represented to the computer-controlled operator response module. The FM has a substantial impact on both individual player station host computer processing requirements and LAN traffic flow.

4.0 RECOMMENDED ACTIONS

The scope of the requirements developed in Section 2.0 is large. The recommendations to address the requirements in Section 2.0 determine an overall program plan for SAF development. Included in this plan are the following activities:

(1) Establish aircrew team training Military Characteristics (MC's)
(2) Relate MC's to on-going UTSS and SIMNET -D activities
(3) Develop integrated design specification
(4) Conduct workstation-level prototyping
(5) Conduct module fidelity and update rate requirements study
(6) Revise SAF prototype fidelity management module
(7) Develop SIMNET testbed at AFHRL for network experiments
(8) Refine SAF testbed design
(9) Develop formal SAF protocol specifications
(10) Establish host hardware characteristics

VICTORY has established a baseline SAF design and performed preliminary computational requirements(se Reference 4). Extensive off-the-shelf modules and associated data bases are available to support the SAF design process. The overall SAF architecture needs to be developed. The requirements established in Reference 4. suggest that a customized parallel processing host computer, capable of an effective 30 MIP's,is required to achieve high fidelity SAF realtime operation at a given manned player station.

5.0 REFERENCES

(1) "NAVAL AVIATION TACAIR/ASW AIRCREW SIMULATOR TRAINING REQUIREMENTS ANALYSIS", CNO Memo: Ser 593D/9U598436, 12 June 1989

(2) "SENSOR AND COUNTERMEASURE MODELING IN DISTRIBUTED SIMULATION",VICTORY Memo: M-90-01, January 1990, P. Thompson

(3) "SIMNET SEMI-AUTOMATED FORCES CREW RESPONSE MODELING", VICTORY Memo: M-90-03, January 1990, G. Smith

(4) "MODULAR THREAT ENVIRONMENT SIMULATION DESIGN DISCUSSION", VICTORY Report: May 31, 1989, L. Jobson

VICTORY
INTEGRATED SYSTEMS, INC.

POSITION PAPER

SIMNET SEMI-AUTOMATED FORCES CREW RESPONSE MODELING

MEMORANDUM
M-90-03

Submitted to:

NETWORK STANDARDS WORKING GROUP

Submitted by:

VICTORY INTEGRATED SYSTEMS, INC.
6120 Paseo Del Norte, Suite O-2
Carlsbad, CA 92008

by

Gregory Smith

January 1990

INTRODUCTION

VICTORY INTEGRATED SYSTEMS, INCORPORATED (VICTORY), is a small business located in San Diego, California with a broad background in weapon systems design, simulation, and training. In particular, VICTORY personnel have extensive experience in simulating aircrew behavior to assess the adequacy of new and modified crew station designs. We have also assisted in the development and testing of aircraft full mission simulator models. This paper addresses the need and issues relating to the development of Operator Response Models within the SIMNET Semi-Automated Forces program.

The goal of the SIMNET Semi-Automated Forces (SAF) program is to develop effective computer-controlled combat vehicle and supporting command hierarchies that can be injected into the combat network. SAF's are required to simulate the large number of battle field participants needed for effective combat team and command training or weapon system development at reasonable cost. The behavior of a SAF unit should be indistinguishable to an observer, or simulation exercise participant, from a manned unit of the same type to ensure that effective training is provided to manned SIMNET stations. This also ensures that weapon systems design concepts can be explored reliably in realistic combat engagements.

A highly detailed operator response model (ORM) is required to ensure that SAF players behave realistically during SIMNET exercises. In effect, the ORM is required to simulate the functions of the combat crew for each SAF player (e.g., tank, rotorcraft, etc.). At the most basic level the ORM is responsible for collecting and processing tactical information, executing SAF controller commands, and engaging or providing support to manned SIMNET stations.

This paper addresses the requirements and top-level methods for the SIMNET SAF ORM. The methods proposed for operator behavior are based on techniques developed for modern avionics situation assessment and tactics planning programs. Extra emphasis is placed on constraining ORM performance using crew workload modeling techniques developed as cockpit design aids. The requirements and methods presented here are based on modeling realistic rotorcraft and fixed-wing aircraft combat behaviors. The requirements for aircraft ORM's are emphasized because the dynamics of engagements involving these types of assets are the greatest. The design constructs presented here are also appropriate for developing SAF tank crew and C^2 ORM's.

II. BASIC ORM MODEL REQUIREMENTS

Several key factors that must be considered in the development of a realistic ORM are the types and timing of responses elicited by SAF players. First, the behaviors produced by the ORM must be representative of those produced by manned combat players in similar combat situations. This means that the ORM requires extensive situation assessment (SA) capabilities and a large data base of potential combat tactics and plans. Second, the observable information that the ORM model uses in making combat decisions must be consistent with that available to a manned system. For example, SAF players cannot be provided information regarding threats outside of sensor range or masked by terrain. Next,

the timing of the SAF player responses must also be correct and reflect all workload demands that would be levied on an identical manned SIMNET player in the same situation. For example, the firing of tank guns must reflect possible crew reloading tasks. Finally, the behavior of the ORM must be able to be influenced directly, or indirectly, by SAF player controllers.,

ORM SA algorithms are required to identify and prioritize SAF player tactical objectives in the context of its overall mission objectives and current environmental conditions. For example, the SA algorithms are responsible for determining which objects are targets and threats and the potential impact of each object on the mission. Algorithms being developed in existing programs such as Air-to-Air Attack Management (A^3M) and Pilot's Associate (PA) can provide a basis for developing ORM SA algorithms.

Crew SA is developed in manned systems by polling four major data sources:

- o External Visuals
- o Sensor Indications
- o Pre-briefed Reconnaissance/Mission Objectives
- o Data Links/Communication

Most tactical SA is developed by the crew from the examination of external cockpit visuals and sensor indications. The ORM must have a capacity for accessing track data from SIMNET SAF sensor simulation components to retrieve these data. Sensor simulations should provide the ORM with all measured and derived state information for each detected track. These data can be used by the SA prioritization algorithms to sort SAF player mission objectives. All sensor system errors should be propagated through the ORM SA algorithms to reflect realistic uncertainties in processing simulated track data by the ORM. The ORM requires a fusion algorithm to incorporate all known tactical information into a single composite track file for ranking purposes. An ORM data base is required to store pre-briefed mission objectives and reconnaissance data. The mission objectives data bases stores basic information for classifying tracks (target, threat, defended asset, etc.), SA track parameter ranking function weights, relative track ID importance factors, and pre-planned tactics. The reconnaissance data includes pre-briefed target and threat locations that can be used by the ORM to influence SA track ranking activities and anticipate upcoming mission events. Requirements for modeling communication and data links are more stressing. Communication models must not only be capable of passing information between SAF players (e.g, target assignments and locations) but must also provide a gateway for receiving data from SAF player controllers. Data received by the ORM from SAF controllers should be used to modify the ORM's top-level tactical priorities and SAF player response plans.

A large data base of tactical responses must also be stored to drive SAF player behavior. This data base should be implemented as a rule-based expert system. A rule-based paradigm is the most practical data structure for storing ORM tactics as it can be modified quite easily by system users to reflect new tactics. Rule-based expert systems also provide trace-back information that can be used validate the ORM decision making process used to generate specific SAF player tactical responses. The rule base can be pre-compiled to

achieve faster run times if the performance of the system is inadequate to keep up with processing demands. The rule base should be partitioned into major tactical segments that are based on top-level tactics options selected by SAF controllers.

A major criticism levied against current computer-controlled threats is that their behavior is very predictable. Most computer-controlled threats are predictable because they rely on deterministic decision making models. Model predictability can lead to negative training as manned players learn to anticipate the responses of computer-controlled player. To alleviate this problem, the ORM rule base should incorporate multiple responses that are appropriate for a given tactical situation. The actual response should be selected randomly from a set choices to make the behavior of SAF players less predictable. ORM response selection probabilities can be specified non-uniformly to reflect the likelihood that specific tactics will be followed.

III. ORM CREW SIMULATION MODEL

The timing of SAF player responses are as important to simulating realistic combat aircraft behavior as the responses are themselves. Accurate and realistic timing of SAF player responses can only be achieved by simulating the actions of the crew itself. The crew simulation can be used to limit the collection of tactical information and response generation to that achievable by manned players. For example, it should not be possible for the SAF ORM model in single-seat aircraft to collect information from heads-down sensor displays and out of cockpit simultaneously.

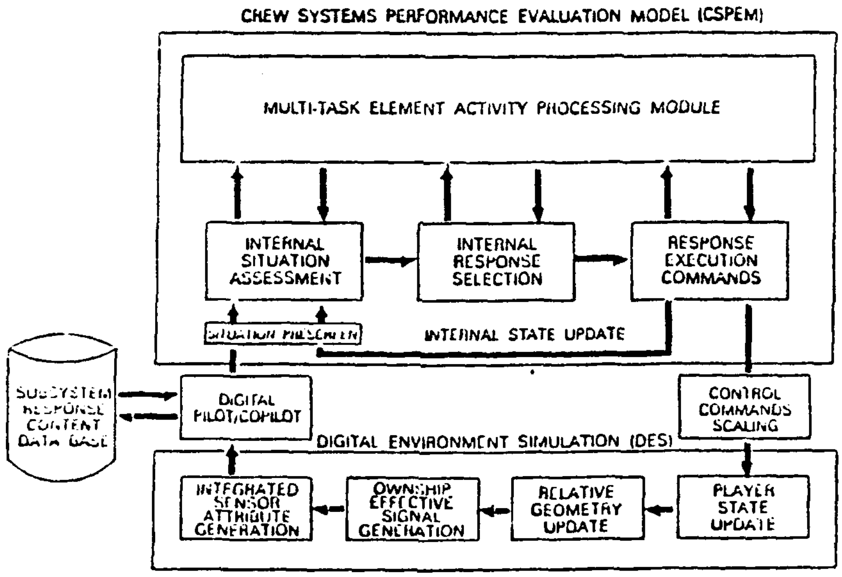


FIGURE 1. CTMEM INTEGRATION OF OVERALL CREW TASK DEMANDS.

Simulations have been developed by the human factors modeling community that are appropriate for modeling SAF crew behavior. These models have been primarily used to predict crew member workload and performance in new crew station designs. The most

successful of these models simulate explicit crew task responsibilities to achieve desired mission objectives. For example, the examination of aircraft instruments to perform a navigation cross-check would be simulated as a set of one or more component tasks. Individual tasks are modeled with completion time and crew resource requirements. Expected crew workload is computed by combining the resources requirements needed to perform all required mission tasking as a function of time. Figure 1 illustrates how the separate tasking requirements combine from major functional crew responsibilities to develop overall task processing demands in VICTORY's Combined Taskload/Mission Effectiveness Model (CTMEM) [1].

The CTMEM (initially developed under AFHRL sponsorship) simulates each weapon system crew member as a server within a queueing system. The CTMEM's queueing system is used to limit the crew's ability to perform required mission tasking based on physical and cognitive crew resource limitations. For example, the CTMEM would not allow the crew to perform two tasks simultaneously that require the use of the left hand. One task would be forced to wait in the queue until the simulated crew member's left hand used to process the other task is available. The queuing system model is also used to ensure that task demands are processed in descending priority order. For example, the CTMEM would simulate crew tasks related to countermeasure deployment prior to those related to navigation systems update. Figure 2 illustrates the functional processing of crew tasking in the CTMEM.

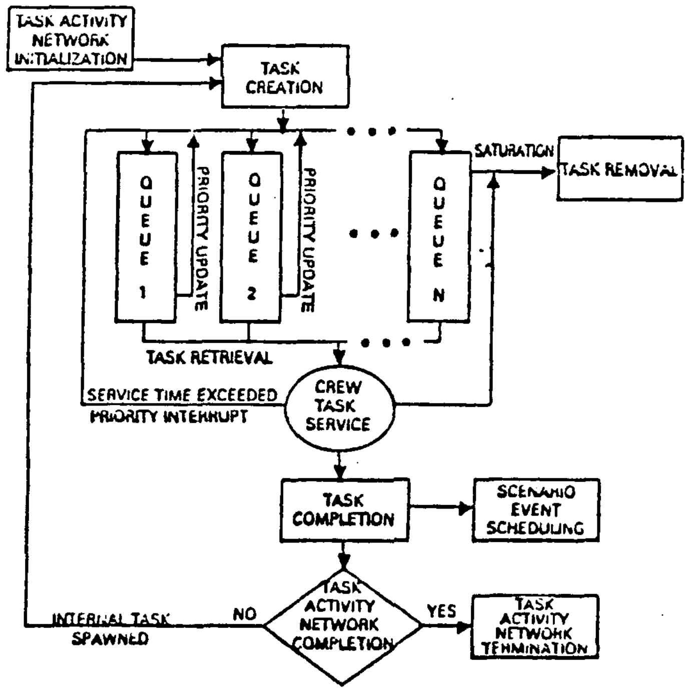


FIGURE 2. CTMEM CREW TASK PROCESSING MODEL STRUCTURE.

The simulation of crew task procedures can be used to constrain and control the flow of information collection and decision making by the SAF ORM. The successful completion of each simulated crew information collection task can be used to incrementally build SA within the ORM in a realistic fashion. For example, new radar tracks would only influence ORM decision making after the display has been scanned by a simulated SAF player crew member. Likewise, SAF players should only fire weapons after all tasking (e.g., weapon select, target designation, missile uncage, etc.) needed to perform the event have been completed. The order in which tasks are processed must also reflect ORM tactical priorities and overall mission objectives specified by SAF player controllers.

Several new SAF data bases would be required to support the existence of an ORM crew simulation. First, a data base of basic crew component task performance would be required. This data base would contain the time and pilot resource requirements (vision, speech, hand, etc.) needed to complete specific crew actions (e.g., switch hits). The crew resource requirements in this data base are used to determine which tasks can be processed by the ORM crew model simultaneously.

A crew task procedure data base is required to link component tasks together. Task procedures would represent functionally-oriented crew behaviors (e.g., instrument cross-check). Task procedure data bases must reflect differences in crew station designs. For example, the tasks needed to place avionics systems to air-to-air attack mode are different in the F-15 and F-16.

A third data base is required to tie crew task procedures to ORM SA, tactics planning, and response execution activities. This data base is the most complicated to define as it interfaces to the general ORM tactics rule base. This data base must also contain the baseline priority for executing all task procedures. These priorities can be used to determine the order of task processing by the ORM crew simulation when resource conflicts occur. The ORM should have provisions for updating these priorities based on derived SA information.

IV. RECOMMENDED ACTION

The SIMNET working group should initiate a study to identify a preferred approach for integrating detailed operator performance models into the SAF system. As a first step, a review of existing crew workload performance simulations should be conducted to determine if an existing model can be used in this application. The review of these models can be used as a basis for updating general SIMNET SAF ORM requirements which can be used to guide development activities.

The existence of core task performance data bases to support ORM crew performance simulations should also be assessed by the working group. These data bases will be needed to establish specific patterns of realistic task behavior for SAF players. Department of Defense computer aided crew station design programs such as the Cockpit Automation Technology (CAT), Advanced Technology Crew Station (ATCS), and Army/NASA

Aircraft/Aircrew Integration (A^3I) programs may provide logical path ways for establishing ORM crew model data bases.

VICTORY looks forward to participating with the Network Standards Working Group to establish specifications for SIMNET SAF Operator Response Models. We have existing software component models that satisfy many of the ORM requirements described above and are continuing to evolve these models in ongoing VICTORY crew station evaluation and simulator development programs.

V. REFERENCES

1. Jobson, L.B., Combined Taskload/Mission Effectiveness Model, VICTORY INTEGRATED SYSTEMS, INC., May, 1989.

24

Distributed Simulators Architecture (DSA)

January 26, 1990

L. Michael Sabo
Martin Marietta Team
SSDS, Inc.
1101 W. Mineral Ave.
Suite 200
Littleton, Co 80120

1.0 Executive Summary

The current BBN proprietary SIMNET protocol is not capable of providing an open and extensible communications environment to support interoperability of heterogeneous distributed simulators. The BBN proprietary SIMNET protocol was developed to support a specific research and development application and as a proof of concept. Hence, there was no need to consider an open architecture. However, as the technology has evolved and other armed services, finding this form of training valuable, now want to integrate their simulators into the environment, the weaknesses of the BBN proprietary SIMNET protocol have become apparent. Recent experiments with interoperability using the BBN proprietary SIMNET protocol have proved to be less than optimal. For instance, specialized hardware protocol converters were necessary to achieve any form of interoperability. This paper address the truth about the BBN proprietary SIMNET protocol and provides an alternative open communications architecture which also preserves much of the simulator to simulator protocol (termed simulation protocol in SIMNET) development which has been accomplished during the SIMNET project.

This paper proposes a new way to look at the problem, to organize our efforts and thus to get us moving quickly towards finding solutions to distributed simulator interoperability issues across the military services. The Simulation Networking community is being told that the current BBN proprietary SIMNET protocol is application layer only, this is a complete falsehood. The association protocol portion is absolutely not an application layer protocol. It in fact spans the transport and session layers. (BBN actually admits this in Report No. 7102, July 1989 page 55; "The association protocol is designed to offer a streamlined composite of the specific transport, session, and application layer services that are required by both the simulation and data collection protocols") It is clear that the BBD approach is in complete and utter violation of the OSI reference model. The BBN proprietary SIMNET protocol locks implementors into non-standard and proprietary transport and session layer services. Therefore, the simulator protocol portion of the BBN proprietary SIMNET protocol can not be decoupled and implemented using other standard datagram transport layer protocols (i.e., UDP or CLNS).

The lack of a formal architectural description for communications necessary to ensure interoperability, as well as administration of networked defense simulators, has prompted this position paper for consideration.

2.0 Introduction

The members of the standards body for interoperability of defense simulations, as a

group have a goal to create a more robust framework, known as an architecture, which will satisfy the current and future needs for networking defense simulators. If the agreed upon standard only satisfies the needs of a single military service then it has failed in its mission. As a result, a recommendation is put forth that a new and more robust architecture, described in this paper, as the Distributed Simulation Architecture (DSA), be reviewed for adoption as the defense simulation interoperability architectural standard.

A formal architectural description means taking a systems approach, where a very large problem is decomposed into smaller, more manageable problems which can be worked concurrently. To some extent this was attempted prior to the Second workshop on Standards for Interoperability of Defense Simulations; however, it was not done based on an overall framework - thus many groups crossed logical boundaries and many issues had to be completely ignored because there was no "shoebox" for them to be stuffed into. The DSA approach not only defines the architecture, but also defines how it is to be managed.

The following paragraphs describe the realities of the SIMNET protocol and defines DSA. Future papers will follow and describe, in further technical detail, many of the topics introduced.

3.0 The Realities of the BBN Proprietary SIMNET Protocol

The BBN proprietary SIMNET protocol is actually a suite of protocols (association protocol, simulation protocol, and data collection protocol) which extends far below the application protocol layer, although BBN has not presented it that way. The BBN SIMNET protocol manual depicts the SIMNET protocol as shown in Figure 1. If this were accurate then BBN could claim (as they do) that the SIMNET protocol is compliant with the OSI reference model. The fact of the matter is that the BBN proprietary SIMNET protocol should actually be described as shown in figure 2.

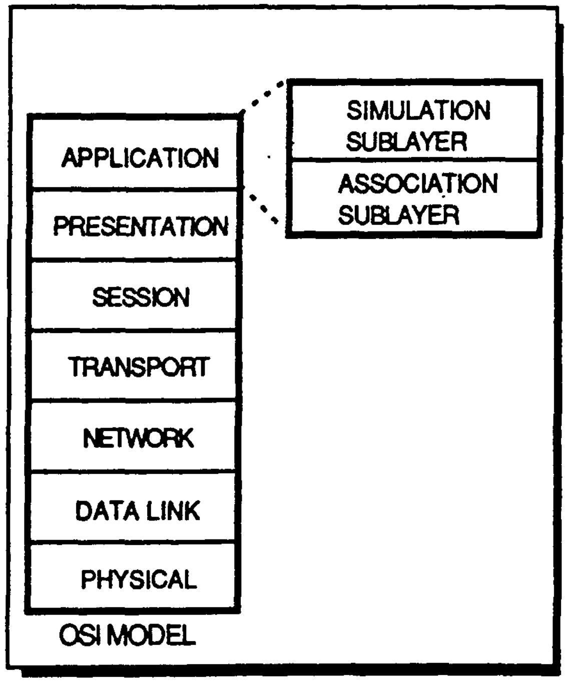


Figure 1 - SIMNET ARCHITECTURE AS DOCUMENTED IN BBN REPORT NO. 7102

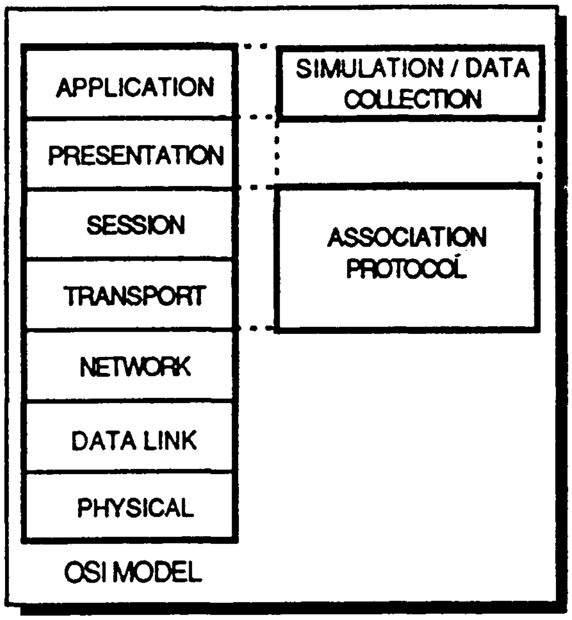


Figure 2 - THE ACTUAL SIMNET ARCHITECTURE

Note that the association protocol is not contained in the application layer. The association protocol is a concatenation of both the transport and session layer services and is completely proprietary to BBN. Thus the BBN SIMNET protocol is a proprietary protocol suite and is in complete violation of the OSI reference model due to its session and transport services being tightly coupled with application services.

Selecting a proprietary protocol suite does not make economic sense. The use of off-the-shelf components will be highly curtailed. As simple examples, how will it handle international simulation interoperability when our NATO allies are based exclusively on OSI protocols? Who makes a router based on association protocol? How many protocol analyzers exist which can analyze association protocol? What open network management packages exist for the BBN proprietary SIMNET protocol? What about security? Clearly the emperor has no clothes.

4.0 DSA Overview

DSA is an open communications architecture for networking large numbers of interactive defense simulators. The architecture will follow the OSI model and will not be limited by network media and will therefore operate over both local area and long haul networks. The architecture will also define a management structure where issues can be addressed and standards developed.

Figure 3 depicts the initial DSA with respect to the OSI reference model.

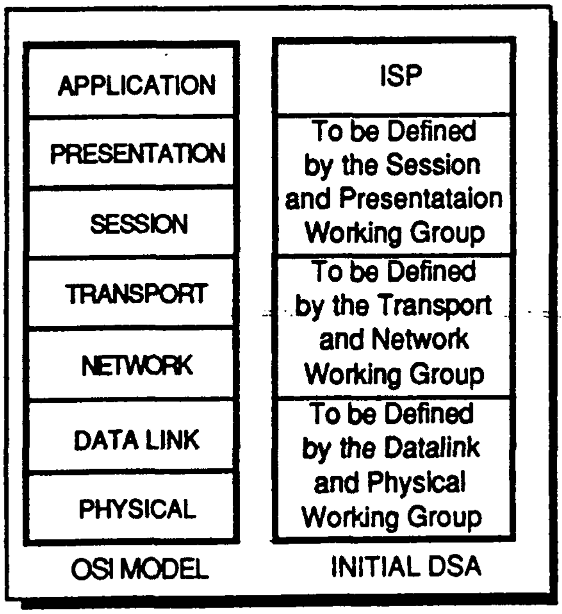


Figure 3 - INITIAL DSA

4.1 DSA Administration Structure

DSA will be managed by the DSA executive committee, consisting of members selected from government, academia, and industry who have an interest in establishing and maintaining a truly open distributed simulation architecture. The chairman of this committee will be known as the DSA architect and his duties will be defined, via a formal charter, by the DSA executive committee. In addition to the executive committee, individual working groups will be established. The function of these working groups will be to address issues of DSA, investigate solutions, and forward their suggestions on to the executive committee. The working groups addressing communications issues will be structured along the OSI reference model. Therefore the following working groups will be established:

- Interactive Simulation Protocol (ISP) Working Group
- Session and Presentation Protocols Working Group
- Transport and Network Protocols Working Group
- Datalink and Physical Protocols Working Group

Other working groups necessary within DSA will include:

- Database Working Group
- Security Working Group
- Network Management Working Group
- DSA to BBN proprietary SIMNET Gateway Working Group

4.2 Interactive Simulation Protocol (ISP)

It will be the charter of the DSA executive committee to develop a military standard application layer protocol which shall satisfy the requirement of passing standard messages between simulators DoD wide. In addition, ISP will be submitted as an international standard. As a true application layer service, ISP, will not be dependent on the protocols which reside below it. It is suggested that DSA use the simulation protocol portion of the BBN proprietary SIMNET protocol as its baseline for this application layer service. However, that decision would be made by the ISP working group and the DSA executive committee.

ISP will not include the data collection protocol portion of the BBN proprietary SIMNET protocol. The open network management protocol (i.e., SNMP, CMOT or CMIP) selected for DSA would be utilized for this function.

Other papers will follow which describes the ISP requirements in depth, the charters for each of the other working groups, and the use of a network management protocol for the data collection function.

5.0 Conclusions

DSA lays the foundation for the open integration and interoperability of heterogeneous defense simulators. BBN will likely argue to protect their proprietary protocol suite; however, a single organization should not have the final say on the matter. It is recommended that the Institute for Simulation and Training adopt DSA as the standard simulation networking architecture in its recommendation to PM TRADE.

6.0 References

Tanenbaum, Andrew; Computer Networks, 2nd Ed.; Prentice-Hall, 1988

BBN Report No. 7102; BBN Systems and Technologies Corporation; July 1989

Standards For Interoperability of Defense Simulations
Long Haul Subgroup

Position Paper On Goals And Issues

David C. Wood
The MITRE Corporation
7525 Colshire Drive, MS W420
McLean, VA 22102-3481
(703) 883-6394
e-mail: wood@gateway.mitre.org

22 January 1990

Current Status

- Simnet is designed to exploit LANs (broadcast, low delay, high throughput - in particular Ethernet)

- Can interconnect LANs in a limited way - e.g., 56 Kbps circuit supports 100 vehicles

- For March demo, will use ST over Terrestrial Wideband Net (TWN), T1 backbone (because its there and provides multicast, service guarantee). ST is an experimental protocol that is not recommended for general use and is not widely implemented.

- Not clear how to scale up beyond single T1 bus

Goals

- Specification of Simnet protocols and interface, so any vendor building a simulator can plug into any standard network and interoperate with other Simnet simulators on interconnected networks, including long haul interconnection of LANs, independent of underlying network technology.

- Simnet protocols should not be concerned about network configuration, although some network configurations may not be practical for some simulated configurations (e.g., close flying aircraft may not be allowed at separate locations on wide area net).

- Simnet protocols should eventually use a profile of OSI protolcols below application layer. If current OSI protocols (GOSIP) are inadequate to support Simnet applications, this must be identified and research and experimentation conducted to develop proposals for appropriate OSI enhancements.

- The Simnet protocols and interface should scale up, without change, to enable simulations involving 10s of thousands of elements (vehicles, troops) at 100s of locations.

- Need to define the network service requirements that Simnet will need, so that future operational DoD networks (e.g., enhanced MILNET, IDCS-WESTHEM, FTS-2000) meets those requirements. Requirements issues include guaranteed service for real time application, multicasting, appropriate security level(s).

General Questions for the Subgroup to Answer

- Is the current PDU specification adequate to meet the goals stated above?
- Are some additions needed to the PDU specification?
- Is the proposed standardization schedule too soon to determine long-haul needs?
- How do we get from where we are today (using an experimental research environment ST over the DRI) to the goal (using standard protocols over whatever operational networks with the required services are available)?

Specific Technical Issues

- Does the current use of Ethernet comply with IEEE 802.2 and 802.3 as specified in GOSIP?
- Could the Simnet protocols make effective use of any of the OSI upper layer services, e.g., connectionless stack, remote operation services, naming services.
- Does TADIL-J exchange similar information to the SIMNET protocols? If so, why couldn't TADIL-J be used in the SIMNET standard, to facilitate interoperability between real and simulated systems? (One potential issue is that TADILS are specific to the communications systems and encompasses all protocol layers.)

- In a Simnet architecture involving LANs connected via gateways to a long-haul network, there are architectural trade-offs. The gateway could be at layer 3 in which the application is transparent; or it could be an application gateway which does some processing of the Simnet application data in order to optimize use of the long-haul communications.
- Does the current Simnet specification make provision for interfacing simulators to other standard LANs (e.g., IEEE 802.5 token ring, FDDI)?
- What steps are required to configure the SIMNET devices participating in an exercise? Can OSI management protocols be used?

POSITION PAPER ON THE SELECTION OF A GLOBAL COORDINATE SYSTEM

by Rich Soeldner
Naval Training Systems Center, Code 281
12350 Research Parkway
Orlando, Florida, 32826-3224
407-380-8202

At the conference on Interoperability of Defense Simulations, on Jan. 15th and 16th, recommendations were made to adopt the use of Geocentric Cartesian Coordinates as the SIMNET standard for global coordinates. I believe careful consideration of the compatibility issue should be made before this recommendation is adopted. This coordinate system has some potential advantage when used with CIG systems which require curved earth representations but it is incompatible with the greater majority of training systems as well as incompatible with the present implementation of SIMNET.

Most training systems use restricted play or game areas and use Cartesian Coordinates defined with a North/South and East/West ground axis and altitude as the normal axis. The advantage of this system is that it is relatively simple to implement and it is relatively accurate. At a range of 300 nm there is less than a .03% difference between the great circle range and the pythagorean range. And the difference decreases as the range becomes shorter. Velocities are also relatively easy to determine in terms of ground based Cartesian components. The problem with the Geocentric system is that there is no apparent way to avoid the overhead associated with continuously having to transform from the Geocentric system to the trainer coordinate system and vice versa. There also does not appear to be any way to use the Geocentric coordinate system directly without incurring significant amounts of overhead because the X, Y, Z physics changes with the position of the gravity vector. Therefore it appears likely that simulations will continue to compute X, Y, Z velocities and positions in a standard ground based Cartesian system then transform them into the Geocentric system. Likewise external vehicle velocities and positions will have to be transformed from the Geocentric system into the standard Cartesian system for use.

There is an alternate system of coordinates which is compatible with both the restricted Cartesian play areas found in most simulators and global coordinates. This is a Topocentric system which maps the center of the game area to a specific latitude and longitude. The y-axis of the game area would lie along the line of reference longitude and the x-axis would be orthogonal to the y-axis at the point of reference latitude. Positions are located by specifying their x and y coordinates as they are orthogonally projected back to the x and y axes. Altitude or elevation is positive up and is referenced to mean sea level. Note that this

coordinate system specification is identical to that for a standard ground based orthogonal Cartesian coordinate system. To obtain global coordinates from the Topocentric Cartesian coordinates the following transformation equations give precise results.

```
LON = LONref +  ARCSIN(SIN(X)/SQRT(1-(SIN(Y+LATref)*COS(X))**2)

LAT = ARCSIN(SIN(Y+LATref)*COS(X))
```

where

```
LON -> Longitude of point
LAT -> Latitude of point
LONref -> Longitude of reference point
LATref -> Latitude of reference point
X = x/Rearth -> x position expressed as arc length
Y = y/Rearth -> y position expressed as arc length
Rearth -> Radius of earth at reference Latitude and  Longitude
x,y -> Displacement distances along the x and y axes in  units
       consistent with those used to specify the radius of the
       earth.
```

Inversely to obtain the x and y distances from specific latitude and longitudes the following transformation equations give precise results.

```
X = ARCSIN(LON-LONref)*COS(LATref)) Y = LAT - LATref +
2*ARCSIN(SQRT((COS(LAT)*SIN((LON-LONref)/2))**2 - SIN(X/2)**2))

x = X*Rearth
y = Y*Rearth
```

For complete compatibility there are two corrections which need to be made to the ground based Cartesian system to make it consistent with the global system. The first has to do with the computation of true heading. In the ground based Cartesian system true heading is measured with respect to the y dimension since the y-axis is aligned with true north. In the Topocentric system, unless the reference latitude is aligned with the equator, the y dimension is aligned with true north only along the line of reference longitude. Therefore if a traveler were to start out in an easterly direction and maintained that direction his true heading would change with distance traveled. The following equation provides the magnitude of that change.

```
DELTA_HEADING = ARCSIN(SQRT(COS(Y+LATref)**2/
(1-(SIN(Y+LATref)*COS(X))**2))
```

The second correction is a velocity correction. Because lines of constant x or y curve slightly as they traverse the surface of the earth, the distance between lines of constant x or y converge as distance from the axis increases. However displacements are measured with respect to the axis. Since the axis distance is slightly greater than the actual distance traveled the axis

velocity needs to be compensated to make the actual position displacements consistent with the true velocity. The correction to the velocities are as follows:

Vyc = Vy/COS(X) Vxc = Vx/COS(Y)

where Vx, Vy are the specified velocity components Vxc, Vyc are the compensated velocity components The interesting thing about these compensations is that for most practical applications they can be ignored. For instance at a range the error introduced by not considering the global compensations is only about .25% for both range and velocity. At a latitude of 45 degrees and a distance of 250 nm from the y-axis the error in true heading is only 1.5 degrees. The fact is that any trainer which uses a ground based orthogonal Cartesian system does successfully ignore these global variations. And most trainers do use ground based Cartesian systems. There is one other significant compensation which needs to be made to the ground based Cartesian system to give it a global representation. The altitude/elevation of distant objects needs to be adjusted to compensate for the curvature of the earth. The following equation provides adequate results for most applications.

zobs = zact - Rng**2/2*Rearth where zobs is the observed altitude/elevation zact is the actual altitude/elevation Rng is the distance to the distant object Note that most trainers which require this compensation already have it built in. It should be emphasized that the Topocentric coordinate system is not a projection of a flat surface onto a spherical surface, and that the transformation equations from ground coordinates to global coordinates is precise. Therefore, a distant trainer which desires to look at the problem in global coordinates versus ground coordinates can do so with no loss of precision. The local trainer however will not be required to do any transformations which are foreign to it's normal operation. In sur ary, compatibility to existing trainer systems should be a major concern when selecting standards. A geocentric coordinate system will require changes to virtually every trainer on the network. The topocentric coordinate system would require very few, if any, changes. There I recommend that atopocentric coordinate system be given serious consideration as the SIMNET global coordinate system.

Interactive Simulation Protocol (ISP)

the Application Layer of the Distributed Simulators Architecture (DSA)

January 31, 1990

L. Michael Sabo
Martin Marietta Team
SSDS, Inc.
1101 W. Mineral Ave.
Suite 200
Littleton, Co 80120

(303) 798-5520

1.0 Executive Summary

This position paper describes the means to an open application layer protocol for the exchange of information between simulators involved in interactive simulation exercises. This protocol, termed Interactive Simulation Protocol (ISP), is proposed as a true application layer protocol, within the OSI complient Distributed Simulator Architecture (DSA) [1]. It is proposed that ISP use the simulation protocol portion of the SIMNET protocol and include additional Protocol Data Units (PDUs), which have been suggested as a result of interoperability tests from programs such as the Navy's Battle Fleet Import Training (BFIT), to establish a baseline of the required ISP PDUs.

Note that ISP would not use either the data collection protocol nor the association protocol portions of the current SIMNET protocol. The data collection function, while necessary during a simulation, is not part of the simulator to simulator protocol and as such will not be discussed in this paper. A forthcoming paper will describe how an open network management protocol will be utilized to support both the network management and data collection function. As for the association protocol, this is actually a transport protocol and can not be included as part of a true application layer service such as ISP. A later paper will be produced which will describe open transport layer alternatives which could be utilized in DSA.

2.0 Introduction

During the SIMNET program, a lot of time and effort went in to defining the simulation protocol portion of the overall SIMNET protocol. Unfortunately this portion of the protocol was also wrapped into a transport layer service called the association protocol. As a result, the SIMNET protocol is unacceptable as a open systems protocol standard. The goals of ISP would be to use the simulation protocol portion of the SIMNET protocol with additional PDUs being included from the BFIT experience and other interoperability tests. This methodology will facilitate a rapid development of the ISP standard. The protocol will be extensible so that new PDUs can be added while also being backwardly compatible with earlier versions of ISP. This methodology is essential because as we continue to learn more about interactive simulation this experience must be capable of being incorporated into ISP. Further, DSA will include a working group which addresses the task of gatewaying DSA to the current SIMNET protocol so that existing SIMNET simulators can be interconnected to DSA.

As a true application layer protocol, ISP will not be dependent on the particulars of the services provided at lower levels of the DSA

protocol suite including specific transport protocols or whether a given physical connection is via a local area or wide area network. This is very important because the selection of other specific lower layer protocols, for DSA, could be worked concurrently with the ISP development thus cutting the overall DSA development cycle. That is a major benefit of the DSA layered protocol approach.

3.0 Distributed Simulators Architecture (DSA)

An overview of DSA has been submitted to the Institute for Simulation Technology (IST), DARPA, and PM TRADE. DSA is a framework which is in total compliance with the OSI reference model and describes a methodology for the development and management of an open architecture to ensure the interoperability of Defense Simulators. The DSA overview introduces ISP. DSA's thrust is to establish an open interactive simulation application layer protocol (ISP) and the utilization of a suite of open protocols to support the lower layer communications services. Figure 1 depicts ISP's relationship to the OSI reference model.

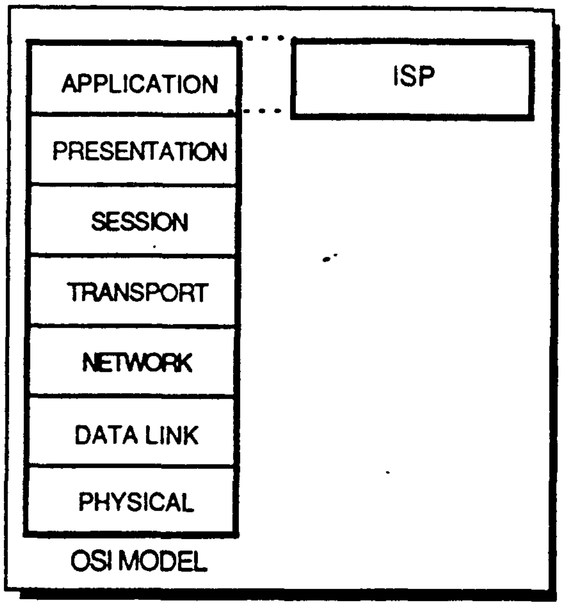


Figure 1 - ISPs Relationship to the OSI Reference Model

4.0 ISP Working Group

It would be the charter of the ISP working group to define the initial version of the ISP protocol. This protocol would be developed based on inputs from the government, academia, and industry. Once complete, the ISP draft standard would be forwarded to the DSA executive committee for review and comment.

4.1 ISP Standardization Timelines

The process of defining ISP as the DoD-wide st^ndard for interoperability of defense simulators should progress rapidly. This would be dependent on the following factors;

acceptance of DSA as the overall framework, establishing the ISP working group, staffing the ISP working group with qualified individuals, and a dedication on the part of the simulation networking community to support this effort. My estimate is that, should the ISP methodology prove acceptable to IST, PM TRADE, and DARPA, a standard specification could be written and a public domain version of the protocol could be developed and tested in four to six months. This estimate is based on the following assumptions; the ISP working group will be composed of six to ten members who are very familiar with the simulator protocol portion of the current SIMNET protocol and the use of this protocol for interoperabilty between dissimilar simulators. Three to five graduate level software engineers with an understanding of data communication protocols would be required to develop the public domain software. Figure 2 depicts this development methodology.

5.0 ISP Requirements

To come to timely fruition, ISP must achieve a number of objectives. This section addresses these general requirements.

- Complete disassociation of the current SIMNET association protocol and data collection protocol from ISP.
- ISP will incorporate PDUs which were developed as part of the simulation protocol portion of the SIMNET protocol, during the SIMNET program. Additional PDUs, which were suggested as a result of BFIT and other interoperability tests with the SIMNET protocol, will also be considered.
- ISP will be extensible such that additional PDUs can be incorporated into subsequent versions while assuring backward compatibility.
- A formal administrative process will be developed where new PDUs and modifications to existing PDUs may be proposed for consideration.
- A working group will be established whose charter will be to certify the interoperability of each ISP implementation.
- No additional PDUs will be allowed to be incorporated into the standard, nor any modification to existing PDUs, without an actual working prototype of such an implementation. This prototype will be used to demonstrate the proof of concept as well as backward compatibility.

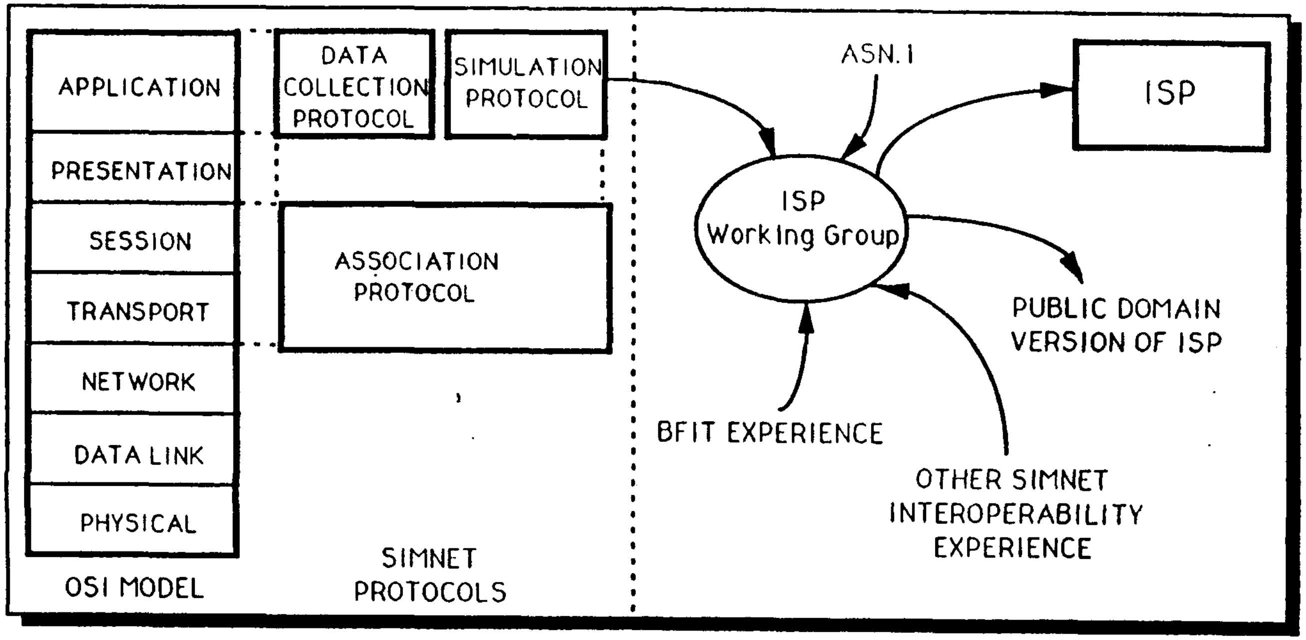


Figure 2 - ISP Development Methodology

- A version of the ISP software will be available in the public domain and on-line. The software should be available on an Internet host which supports file transfers. This ISP implementation could, perhaps, be developed by University of Central Florida graduate students. The public domain version of ISP will serve as a reference implementation for any group developing their own version. In addition, having a version available on-line and in the public domain proves that ISP is an open protocol.

- ISP will use Abstract Syntax Notation One (ASN.1) representation to describe the PDUs in accordance with ISO 8824.

- ISP will specify the use of ASN.1 for the encoding of the ISP PDUs onto the physical media in accordance with ISO 8825.

- The ISP specification will be written into an Internet Request For Comment (RFC) and distributed to the Internet community for review and comment prior to it being submitted as a proposed MIL-STD. The feedback from the Internet community will provide a final sanity check for the protocol.

6.0 Abstract Syntax Notation One (ASN.1)

It is strongly recommended that ISP use the OSI standard ASN.1 to both define the PDUs, as per ISO 8824, as well as use the Basic Encoding Rules (BER) for encoding these PDUs on physical media, as per ISO 8825. The use of ASN.1 ensures that the PDUs being transferred are machine independent. In addition, an ASN.1 parser is available as part of the ISO Development Environment (ISODE) from the University of Pennsylvania. The current SIMNET protocol uses Data Representation Notion (DRN) which is unique to the SIMNET protocol and is not a recognized standard.

In his book Computer Networks, Andrew Tanenbaum [2] writes, "The key to the whole problem of representing, encoding, transmitting, and decoding data structures is to have a way of describing the data structures that is flexible enough to be useful in a wide variety of applications, yet standard enough that everyone can agree on what it means. As part of the OSI development work, ISO has devised such a notation. It is called abstract syntax notation 1 or ASN.1 for short"

7.0 Conclusions

ISP is truly an open application layer protocol which will ensure the interoperability of defense simulators. It is proposed that ISP use the simulation portion of the current SIMNET protocol with additional PDUs being included as a result of the knowledge gained during BFIT and other interoperability tests. It is recommended that the Institute for Simulation and Training adopt both DSA and the DSA open application layer protocol, ISP, in its recommendation to PM TRADE.

8.0 References

[1] Sabo, L. Michael, Distributed Simulator Architecture (DSA); Position Paper for the Second Workshop on Standards for Interoperability of Defense Simulators, 26 JAN 90

[2] Tanenbaum, Andrew; Computer Networks, 2nd Ed.; Prentice-Hall, 1988

NETWORK DESIGN CONSIDERATIONS FOR THE SIMNET PROTOCOLS

Dr. John R. Doner
Government Communications
Systems Division
Harris Corporation

P.O. Box 91000
Melbourne, FL 32902
Ph. 407-729-7083

TABLE OF CONTENTS

1.0 EXECUTIVE SUMMARY

The SIMNET project is a research and development effort undertaken by DARPA and the United States Army to enable the linking of remote training simulators. In this manner, personnel training can be extended to include a wide variety of battlefield functions, with related personnel being able to participate simultaneously in a single battle simulation even though they are at sites remote from one another. With the anticipated decrease in the defense budget, the SIMNET project can permit the US military to maintain readiness at a much lower cost than is possible by use of field exercises. SIMNET is therefore a project of great importance to the future defense posture of the United States.

SIMNET has reached a prototype stage where two computers with separate simulation software communicate with each other over a single Ethernet, demonstrating the feasibility of SIMNET in the simplest possible configuration. While SIMNET encapsulates simulation status information for transfer to remote simulation systems, it does not become directly involved with control of the network(s) which transmit the simulation information. Clearly, a real-time simulation, such as SIMNET will support has timeliness constraints associated with delivery of its data.This white paper addresses the topic of the networks carrying SIMNET data, especially as to whether or not SIMNET data can at all times be expected to arrive at destination stations in a timely manner. The case is made, supported by substantial research evidence, that there are types of traffic and loading possibilities that will not be supported if SIMNET only relies on unprioritized message service from the underlying networks. The scope of SIMNET applications would thereby be narrowed, perhaps preventing the simulation of close air support and battlefield communications functions in all but the smallest scenarios.

Organization of this white paper is as follows. Section 2 discusses the SIMNET concept in more detail. Section 3 reviews certain research results concerning Ethernet and related CSMA/CD network architectures. Section 4 develops some SIMNET scenarios in which timeliness of some types of traffic are quantified. Section 5, drawing on the previous two sections, demonstrates the likelihood that SIMNET, as currently envisioned, might fail to deliver some types of traffic in a timely manner. Section 6 recommends a plan of investigation related to ensuring the timely delivery of traffic on SIMNET. Section 7 is a summary of Harris Corporation credentials in the area of network communications and design, and Section 8 is a list of the references cited in developing the arguments of this paper. Although the results of this paper cannot be based on actual SIMNET traffic loadings, they indicate that it is time to examine the network architectures which must support SIMNET to assure the widest possible scope for the system.

2.0 INTRODUCTION

This section will discuss the intended use and scope of SIMNET, and will identify some areas of research that may be of importance for the long-term feasibility of SIMNET in practical large-scale applications.

2.1 SIMNET and Distributed Simulation

SIMNET is a research and development project sponsored by Defense Advanced Research Projects Agency (DARPA) and the United States Army, the purpose of which is to define and validate protocols by which separate training simulators can be interlinked over existing communications networks. The intent of this activity is to enhance the value of training simulations by permitting multiple combat training functions, normally carried out at physically separate facilities, to be combined into a single training exercise.

For example, personnel being trained to crew tanks in land battle could be trained in a scenario where many other components of the battle were realistically represented by human operators at sites remote from the tank crew trainees. The tank crew trainees might find themselves working in concert with close air support elements, supply vehicles, field artillery, etc., and, of course, adversary battle elements could also be represented.

The mechanism by which training simulators remote from each other might be combined would be the set of information protocols developed by the SIMNET project. I.e., the participating simulation software in a SIMNET-linked joint training exercise would encapsulate state information about the local simulation participants for propagation to all other participants. Similarly, a local simulation would receive and properly process all of the relevant state information from remote simulation participants.

The remote state information received at each location would be used to update information concerning the status of the remotely simulated battle elements. All simulators participating in the joint exercise would have access to a common terrain data base so that the received simulation data could be used to properly position and orient remotely simulated elements within each local visual representation of the battlefield. Other important visual characteristics of battle concern the presence of visual obscuration (smoke, fire, etc.) on the battlefield, and the condition of the battle elements (on fire, destroyed, turreted gun in motion, etc) participating in the battle; this sort of data would also be passed among simulators by the SIMNET protocol.

The SIMNET protocols have been designed to identify and abstract the important characteristics of each battlefield element, so that such information can be encapsulated in standard data packets of various kinds for network transmission. Any given type of data packet contains standard data fields of fixed sizes into which the required data is inserted. Such packets can then be sent to all other participants in the simulation for which the particular data is relevant. Some transmitted packets may be relevant to all other participating simulators, such as a status report on the current location and motion of a vehicle. Some transmitted packets may be of interest to only a proper subset of all other participants, such as a packet which notifies a specific set of vehicles within the simulation that they are currently being scanned by a radar: this would permit activation of radar warning devices, as well as commencement of countermeasures.

It should be emphasized that the SIMNET protocol structure is not intended to supply a network control structure directly acting on physical network hardware. In fact, SIMNET is effectively an information-encapsulating protocol written at the application level of any network on which it is to be implemented. It can affect or control network operation only to the extent that an application program can access network functions at the lower (network) protocol levels.

The SIMNET system has been demonstrated in protoype version on a standard Ethernet, and alternative network protocols that have been suggested by the developers include Fiber Distributed Data Interface (FDDI), the DOD Internet Protocol (IP), the DARPA Internet Stream (IS) protocol, and the emerging OSI connectionless protocol extended to support multicasting (see references [1] and [2]).

2.2 The Networking Requirements of SIMNET

Because SIMNET does not contain network control functions in any direct or substantial sense, it is import to analyze the suitability of potential existing network architectures for SIMNET use. The remainder of this white paper addresses this topic. Specifically, a case will be made that SIMNET, once enhanced with the full range of desired capabilities, such as accommodation of voice traffic, command and control functions, and high speed vehicles (e.g., close air support aircraft), may require the support of specialized networking functions, such as traffic prioritization.

The arguments presented in this paper are drawn from analysis of potential SIMNET scenarios, and are supported strongly by research results drawn from the papers cited in the References (Section 8.0). Since SIMNET has not yet been applied to any large-scale simulation exercises, what is adduced in this paper about future network loads due to SIMNET traffic is hypothetical. However, the hypothesizing is quite conservative, and at least indicates that serious consideration should be given to an extended analysis of the networking requirements of SIMNET. The analysis provided in this paper strongly implies that the SIMNET system itself should be simulated, in order to determine whether or not actual loading conditions on a combination of networks can be met without more extensive network control capabilities than are now available through the SIMNET protocols.

Note that the purpose of this paper is not to identify deficiencies in the SIMNET protocols or to indicate that the protocol effort currently underway is misdirected. Harris Government Communications Systems Division (GCSD) personnel attended the Standards for the Interoperability of Defense Simulations conference sponsored by DARPA in Orlando, Florida on 22 - 23 August, 1989, and were impressed by the potential scope and practicality of the SIMNET concept (see reference [3] for a summary of the proceedings of that conference). However, Harris GCSD personnel have been involved in many complex network design projects, and therefore recognize that in order to achieve the broadest scope and benefit from the SIMNET effort, the government should initiate a parallel effort investigating the implications of network control techniques and how the SIMNET system might optimize its use of network resources.

3.0 THROUGHPUT/ DELAY CONSIDERATIONS FOR ETHERNET

The current prototype SIMNET implementation is supported by Ethernet communications, and Ethernet is regarded as a good communications basis for SIMNET simulators at a single physical site. This section will discuss certain general and very relevant research done on Ethernet implementations. The results used in this section are taken from open-literature sources readily available in the references. Although these results are not based on exact traffic loading profiles or network configurations which might arise in SIMNET scenarios, they are theoretically close enough to some possible SIMNET scenarios that they cannot be dismissed as irrelevant.

3.1 Mixing Voice and Data on Ethernet

It was mentioned at the Orlando SIMNET conference (August 22-23, 1989) that the SIMNET system eventually intends to support voice traffic and command and control functions. In fact, Dr. Jack Schwartz, (Director of Information Science and Technology Office, DARPA) specifically noted that simulation of communications was a very desirable goal for the system in the opening address delivered to the conferees. In order to provide an intrinsic capability for voice traffic in SIMNET, it will be necessary to deal with the need to forward digitized voice packets using the underlying network capabilities supporting SIMNET. Because packetization of voice places special demands on a network, the feasibility of this concept must be analyzed in some detail.

Voice messages travel on digitized networks by being broken into many small packets, since a single complete utterance can easily require hundreds of thousands of bits. These individual packets must traverse the net and then be reassembled at the destination before being converted to analog form. Voice packets must be handled in a very timely manner, since the rate of "readout" at the received end must closely match the real-time rate of the original spoken text. This does not require extremely short delivery times, since a modest delay of a voice communication may not substantially affect the outcome of the simulation, but it does require that the voice packets arrive without too much variation in delay. Achieving this low variance in delay may be easy on a very lightly loaded network (say 5% of bandwidth in use), but it will become more difficult as the traffic load grows.

It is instructive to consider a 10 Mbit/second Ethernet implementation with a linear bus carrier sense multiple access/collision detection (CSMA/CD) scheme as a typical present day supporter of a potential SIMNET simulation. In a CSMA/CD Ethernet, access to the bus is obtained effectively autonomously: any attached station with traffic to transmit first determines if there is activity on the bus, and if not, transmits the queued traffic. This system is not perfect, because of propagation delay. I.e., two stations may both sense that the channel is inactive, and may both begin to transmit. This need not actually occur simultaneously, since the transmission from an already transmitting station may not reach a second station in time to preclude initiation of transmission from that station.

When such a simultaneous attempt at transmission occurs, the involved stations can detect it (by comparing their transmitted bit streams with the apparent signal on the bus), and a collision is said to have occurred. Once a collision has occurred, both of the packets in transmission are lost, and both must be retransmitted. The details of this process are well described in [4], where it is also shown that in the traditional Ethernet CSMA scheme, the inefficiency of the CSMA/CD channel access protocol results in a maximum possible throughput of approximately 32 % of channel capacity. For a 10 Megabit Ethernet implementation, then, a maximum sustained throughput of at best 3.2 megabits/second of data can be expected.

There are various strategies known by which the stations involved in a collision attempt to recover from the collision. Obviously, if they both immediately tried to transmit again, another collision would occur, wasting even more channel capacity. Thus algorithms, known as backoff algorithms, are used by which each station randomly selects a time delay until making the next attempt at transmission.

Now we will consider that capacity in terms of digitized voice transmission techniques. In order to support 64 Kbit per second voice digitization, a 320 bit packet could be sent every 5 milliseconds. Packet overhead must also be accounted for, which drives the packet size up to 536 bits (these values are taken from reference [5]). If we assume half-duplex voice links, then each such virtual voice channel on Ethernet would require 107.2 Kbits/second of data bandwidth.

The research described in [5] specifically examined the results of mixing voice and data on a 10 Mbit/second Ethernet implementation. In that implementation, the total load comprising all non-voice data was set at 5%, and voice conversations were set at 30 - 50 conversations. (Strictly speaking, this exceeds the 3.2 megabits/second of actual bandwidth available, but that is because the voice conversations are not continuous, and the backoff algorithms investigated were not standard Ethernet algorithms.) This was considered to be representative of an ordinary technical research environment use of an Ethernet, based on the Xerox Palo Alto Research Center. Although the data versus voice proportion of traffic, and the total network loading in [5] may not reflect any potential SIMNET scenario, the results in [5] are important, since they indicate that robbing Peter to pay Paul (i.e., delaying data packets to meet timeliness requirements for voice packets) might cost more than Peter can afford. Summarized very briefly, [5] indicates that when a backoff algorithm is used in the mixed voice/data environment which favors voice, then the data packet delay is driven very substantially upward. Specifically, the data packet delays were all in the 100 millisecond range when the backoff algorithm was biased enough toward voice packets to provide approximately 99% on-time deliveries of voice packets.

3.2 The Effect of Station Distribution on Ethernet Performance

The CSMA/CD contention-based access scheme which supports Ethernet communications does not necessarily provide all stations on an Ethernet with equal service. E.g., stations located along a linear Ethernet bus may find themselves more often in contention for the channel if they occupy positions near the end of the bus rather than more central positions. There is a straightforward explanation of this phenomenon: namely, when a station begins to transmit, there is a time window of vulnerability centered on this initiation time during which any other initiated transmission will cause a collision. That window stretches backward and forward in time from the initiation of transmission by the total propagation delay for the station to the extremes of the network. Thus, stations near the center of the bus have windows of vulnerability only half that of the stations near the end of the bus.

Careful research and simulation results on this phenomenon are reported in [4]. In that paper, the authors examined several possible distributions of stations along a linear CSMA/CD Ethernet bus, including uniform spacing along the bus, and various forms of balanced and unbalanced clusterings of stations. Quoting from the authors' abstract in [4],

"...Individual station performance varies with the location of the station. Unbalanced distributions can lead to large performance differences between individual stations with isolated stations achieving relatively poor performance compared to the average...."

In particular, even for a uniform distribution of stations along an Ethernet (this means physically uniform relative to the propagation length of the bus) and a midrange loading of the network,

stations near the center of the bus achieved twice the throughput of stations at the extremities. Using the well-known Utilization Law (page 42, [6]), one concludes that packet delay at extreme stations averages twice that of central stations, if all stations have the same utilization.

In clustered arrangements, stations are not evenly spread across the bus, but are grouped at points along the network. In such cases, uneven cluster size leads to disparate throughputs: this is because the stations within one cluster have small windows of vulnerability relative to their fellow stations within the cluster, but large windows relative to stations outside the cluster. Thus, by way of example, three clusters of stations, with 20, 10, and 10 stations respectively along a 2000 meter Ethernet resulted in about a four to one throughput ratio for the 20 station cluster and the extremal 10 station cluster.

Note that the prototype SIMNET system comprises only two computers on a single very short Ethernet. Each of these computers therefore has a very low window of vulnerability, and the computers get the optimum, i.e., equal service from the Ethernet bus. Simulations on this configuration will be highly optimistic relative to the throughputs and delays to be expected in SIMNET applications.

In most Ethernet implementations, one could expect a clustering of stations, since the bus itself might run between buildings, or between grouped users in several separated parts of a building. In the case of SIMNET applications, where physically large simulation equipments might be required to accommodate the desired training realism, one could expect a clustering of components. Therefore, it may be judicious to assume that message delay for individual SIMNET stations may vary by a factor of as great as four from optimal.

3.3 Possible Concerns About Ethernet Support of SIMNET

The above subsections have shown that

1. data packets in a mixed voice/data Ethernet implementation might require delays in the 100 millisecond range to accomodate the timeliness requirements of voice,

2. stations along an Ethernet bus do not receive equal service from the CSMA/CD access protocol, so that one can expect delay ratios between best and worst service along the bus of as much as four or more.

Although these results are not driven by specific SIMNET scenarios, they do follow from research which is not greatly out of line with some possible SIMNET simulation scenarios. In the following section, we will examine specific SIMNET functions which would appear to require the maximum update rates to be needed in simulation scenarios.

A final note about station distribution problems is that they do not occur only for Ethernet. A paper on FDDI (see reference [7]), indicates that for FDDI token-ring configurations, under some conditions, some stations are almost blocked from the possibility of transmission. This is despite the fact that a token-ring passing access protocol, which avoids contention, is in use. This research indicates that if the FDDI network reaches a near-saturation level for any length of time, some stations, due to their position in the ring, may be unable to transmit for the duration of the period of the heavy load.

4.0 SIMNET MAXIMUM PERFORMANCE CRITERIA

In this section, several specific types of SIMNET applications will be discussed, and some analysis will be presented which implies that the SIMNET system may need the support of a message prioritization scheme in the underlying network protocols.

4.1 SIMNET Interaction with the Underlying Networks

The SIMNET protocols are to be defined so that the underlying network structure can be transparent to them: thus, the desire is that a simulation software system designed to support the SIMNET protocol will effectively be able to cooperate with any of the common network control protocols. Apparently the SIMNET protocol structure would only interact with the underlying network control protocols in the matter of packet addressing/routing.

It also should be pointed out that the nature of a SIMNET-supported exercise might involve traffic transmission over a concatenation of networks. Thus, multiple copies of the same simulation software might be interlinked at a common site over an Ethernet bus, representing the crews of several tanks participating in an exercise. At a remote site, multiple copies of an artillery training simulator might also be interlinked on a local Ethernet bus, and the two aforementioned Ethernet busses might then need to be linked by a long-haul network of some kind in order to make the actions of all participants known to eachother. Higher-level command and control activities coordinating the different types of battle elements might be simulated at yet another site, necessitating the need for another long-haul connection between this third site and both of the lower-echelon sites. Figure 1 provides an illustration of such an internetted simulation.

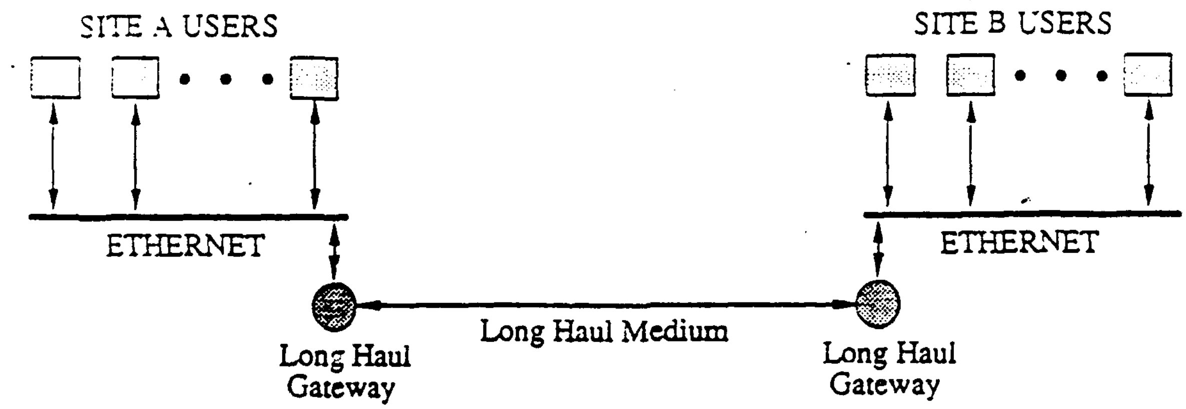


FIGURE 1 -- Remote Internetted Simulation Systems

The mechanisms by which such internetting will be accomplished are not defined as part of the SIMNET protocols: it is assumed that the local facilities participating in a SIMNET-supported simulation have the required physical assets and communications capacity so that the total volume of SIMNET traffic can be adequately handled. Obviously, the degree of success for any proposed internetworking depends very critically upon the total volume of traffic (SIMNET and all other) to be carried and the acceptable delays for the various types of SIMNET data packets in the system, as well as the statistical variance of delay.

For example, it is obvious that the position/orientation data for slowly moving vehicles in the simulation does not need to be updated as often as for high-speed vehicles, such as attacking close air support aircraft. SIMNET takes account of this dichotomy by permitting the information update rates in the networked system to vary according to vehicle type. SIMNET also supports the extrapolation of position information for remotely simulated vehicles by including velocity data in the vehicle status packets. However, it does not support as adequately the extrapolation of orientation information, since it does not include a velocity vector for the pitch, roll, and yaw components of a vehicle. While this may be a negligible oversight for slow moving vehicles confined to surface movement, it may lead to serious ambiguities for aircraft, which might move from a current to a new orientation by one of several trajectories. Clearly the rate of real updates possible has much to do with whether or not such trajectory ambiguity could arise: we will return to this possibility after further analysis has been presented.

SIMNET does not actually define the precise algorithms by which such extrapolation is done: that is left to the individual simulation developers. In any event, a SIMNET-based simulation may be capable of "real" update, based on newly received data, as well as "virtual" update, based on data extrapolation related to the last data packet received concerning the vehicle of interest.

In order to obtain a good basis for a required update rate, let us consider what an acceptable visual display should involve. The motion seen in ordinary US television is the result of a 30 field per second update rate, where a field is a completely refreshed picture. European television standards rely on a 25 field per second update rate, which many Americans at first find uncomfortable, but generally can adapt to. However, when field update rates fall much below this, the extended viewing can become quite uncomfortable. Thus, it is safe to postulate about a 40 millisecond (25 fields per second) update rate as a lowest acceptable rate for extended use by viewers.

Note that in the SIMNET context, this does not necessarily imply that each vehicle being simulated have a real update each 40 milliseconds: the updates may be virtual (i.e., extrapolated) for some period of time, with real updates at longer intervals to pull the virtual updates back toward the precise track of the vehicle (see Figure 2). However, there are some types of data that SIMNET should accommodate for which the virtual update process may not be adequate. These types of data will require better network service than might be available in a network with non-prioritized traffic. These data types will be briefly introduced below, with reasons provided as to why they may require specialized network service.

4.2 Accommodation of Close Air Support Functions

The intended purpose of SIMNET is to enable simultaneous training of multiple battle elements in realistic battlefield context: this purpose will be inadequately served if the very important factor of close air support (both offensive and defensive) cannot be accommodated by the system. The distinction between close air support vehicles and ground vehicles is, of course, that close air support aircraft are capable of very high speeds, and can be expected to undertake high acceleration, unpredictable maneuvers as part of their defensive tactics. Clearly, such vehicles would require a greater ratio of real position updates in comparison to virtual updates than would slowly moving ground vehicles. It is worthwhile to examine the possible situation quantitatively.

As a rough guide to the need for updates, note that an A-10 can turn inside a 1950 ft. radius circle, at a turn rate of 25 degrees per second, and is expected to use this high maneuverability in

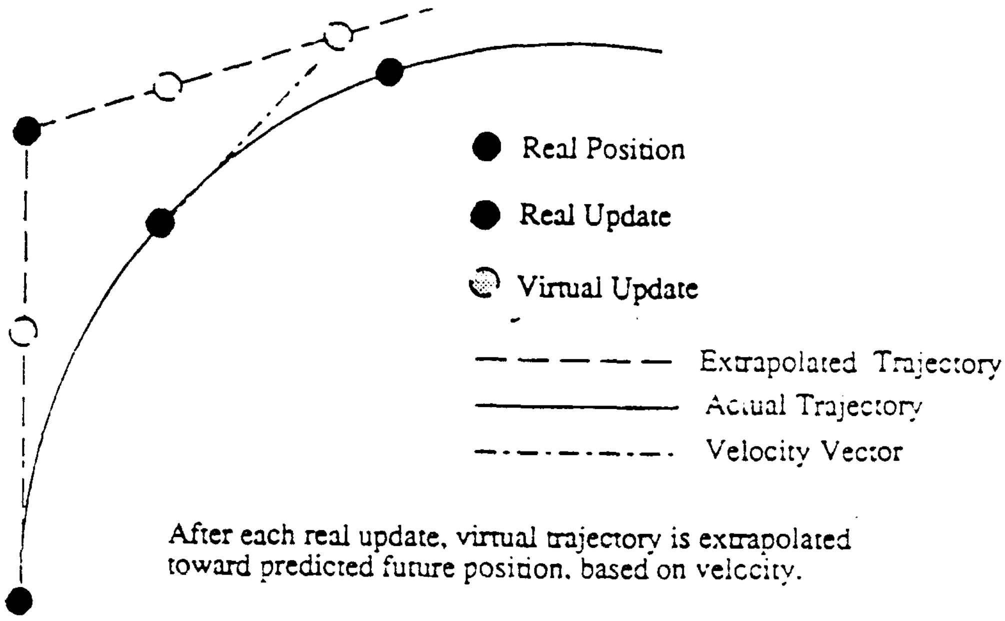


FIGURE 2 -- Use of Virtual Updates to Extrapolate Vehicle Trajectory

evading enemy fire while engaging in offensive actions close to the ground. Such a turn would be made at 250 knots groundspeed and approximately an 80 degree bank angle, and 5.5 G's acceleration. As shown in Figure 3, straight and level flight at this speed covers 422 feet per second. Commencing a turn as described, will result in a deviation from the straight and level path of 92 feet in 1 second, or about 25 feet in 1/2 second. Of course, it takes some time to roll into such a turn, but the result remains true when already established in the turn: namely, if virtual updating of the A-10's position was being done for 0.5 second from the last velocity vector given (a tangent to the turn circle), the extrapolated position and the actual position could differ by 25 feet, as shown in Figure 3.

In actual combat, the A-10 may carry out several relatively quick "jinking" maneuvers made up of combined rapid rolls in opposite directions, as well as rapid ascents and descents which would create substantial changes in forward airspeed. One tactic to evade enemy targeting is to deliver ordinance, and then descend behind trees or other obstructions that provide cover from direct enemy targeting. It is clear that any interplay of tactics between antiaircraft measures and close support aircraft will require simulated distance accuracy in the neighborhood of 10 feet, since an A-10 displaced upward by ten feet from its actual position may appear to be visible to enemy gunners or antiaircraft elements when in fact it is not, thereby allowing the enemy targeting process to have ample time to target, lock, and fire when perhaps in reality the window of opportunity needed did not actually exist. Based on the discrepancy described above for an A-10 in a tight turn, a real update would be needed on the order of every 0.25 second to hope to achieve an extrapolated trajectory for the A-10 which stays within approximately 10 feet of its actual position.

Note also that the interplay of antiaircraft and close air support tactics may require very accurate aircraft orientation, if the weapon being targeted against the aircraft is radar or infrared guided. In such cases, the pilot's maneuvers may be designed to present the antiaircraft crew

with the least desirable (for their weapons) orientational aspect of the aircraft. In such a case, a faithful update process of orientation at very short intervals may be necessary to accurately decide whether or not a given weapon has achieved the required lock on the aircraft. Flares and chaff may also be dispensed by the aircraft, and precise timing of that data may be very critical to the decision as to whether or not a very fast moving maneuverable missile hits its target. It would be unfortunate indeed if the simulator supporting anti-aircraft training declared a hit while the simulator supporting pilot training declared a miss.

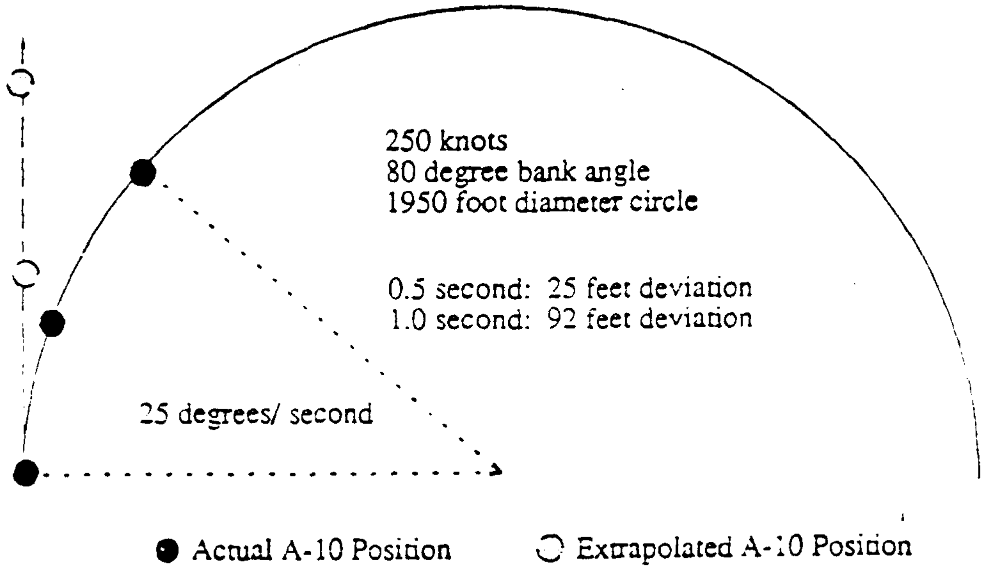


FIGURE 3 -- Deviation of A-10 Trajectory During High Acceleration Maneuver

We have thus established by some relatively straightforward analysis that the possibility exists that support of close air support aircraft in a joint simulation scenario could require real updates of vehicle position/velocity at about 0.250 second intervals. We will consider this requirement again in section 5.0.

4.3 Command, Control, and Communications Simulation on SIMNET

Besides the mere inclusion of voice on SIMNET as part of ordinary command, control and communications (C^3) capabilities, there are other aspects of C^3 functions which might require special consideration. But before discussing these aspects in detail, it is important to examine the degree of fidelity which might be desired for C^3 functions within a SIMNET simulation.

The SIMNET project is directed toward tying together training simulators, not research and development simulators. Therefore, it is reasonable to assume that primarily, C^3 functions would be included in SIMNET to aid in training personnel in proper C^3 techniques. For this purpose, the fidelity of C^3 functions within the SIMNET structure would need not be modeled to a great level of technical detail: so long as the C^3 system appears correct to the end-user, the level of modeling is adequate.

Using this criterion, we can then build requirements from the top down. First we must ask what major aspects of C^3 activities must be visible to the trainee to provide a useful level of training in C^3 proficiency. Primarily, it would seem that these aspects are

1. proper identification/formulation/formatting of C^3 information,
2. proper utilization of communications equipment and procedures,
3. proper recognition of and response to attacks (usually electronic) on the C^3 system.

Do any of these aspects impose special requirements on SIMNET? The first would appear to be a function of the user, not the system, so should not require any specialized SIMNET functions. The second, however, may very well require special attention to timing concerns, because in many battle situations there may be communications subnets established based on a common channel, and access to that channel may effectively be based on contention. I.e., a common voice broadcast channel may be a simple AM channel, in which two transmitters active at once badly degrade each other's transmission, or an FM channel, where the first transmitter effectively captures the channel, and any other simultaneous transmission is not heard at all.

Thus, if such transmission processes are to be adequately represented, they must account nearly exactly for the arrival times of packets representing transmission on the channel. Otherwise, the contention of common channel communications will be completely ignored, which will lead to very unrealistically high estimates of the quality of communications during battle. An implication of this is that trainees would likewise misuse, or perhaps overuse communications assets during training if the assets are not limited by realistic constraints.

What might be done in software to account realistically for channel contention? One approach would be for the simulator originating a transmission to time tag each packet of that transmission with its own "real time", so that all other recipients can compare transmissions competing for the same channel in overlapping time periods, and award the channel to the winning transmission (if the channel is FM), or create the interference effects expected for an AM channel. Because of delay and variance of delay in the SIMNET network communications, a simulation receiving communications packets would have to hold each one for some duration (say 0.25 second) to insure that a competing packet with an earlier time tag had not been sent, but which had sustained more delay on the network. (Actually, clock skew between nodes, as well as delay variance in the network, might double or triple this holding time.)

But the simulation software would have to be even more clever, because each transmission might consist of numerous packets, and two competing transmissions might arrive in an interleaved fashion, with network delay variance resulting in the interleaving not reflecting the true order of competition for the packets, as shown in Figure 4. Deinterleaving these, and accounting for the (simulation) network delay and the correct interference effects could clearly necessitate rather complex software algorithms. Prioritizing all such communications packets and sending them as high priority traffic throughout the simulation network would tend to cut down on the delay and delay variance, so that the problems of the destination software would be made simpler, but would not be eliminated. Furthermore, it is unfortunate that packets representing communications which are the losing competitors for a channel must be propagated throughout the (simulation) network when their information value is to be ignored. This represents inefficient use of a network which might already be taxed to handle the continuous simulation load.

But there is a means by which such complexity and inefficiency can be avoided: that method would require each simulator which is originating a sequence of communications packets representing use of a contention communications channel to send a notice packet to the relevant addressees that a communication on the designated channel has begun at a designated time. These notice packets would be very short, and if they were sent throughout the simulation

network as priority traffic, then each affected simulator would receive all notices of contention with much less time skew, and could rapidly choose the winning contender from the notice information. Similarly, all losing contenders would receive the winner's notice, and would recognize that they were locked out of the channel. Thus they could suppress the transmission of all remaining communications packets until the winner of the channel sent a similar notice indicating that the channel was free.

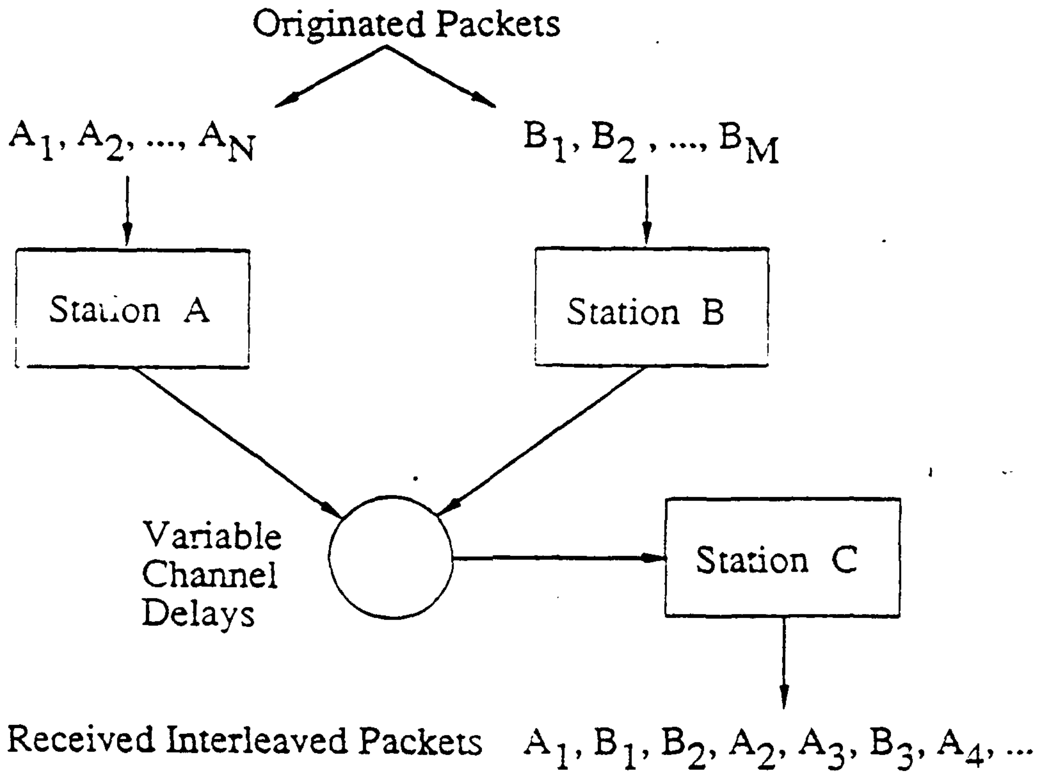


FIGURE 4 – Effect of Variable Channel Delays on Contention Channel Traffic Packets

This technique would greatly simplify the representation of communications throughout the simulation network, and would also contribute noticeably to network efficiency. If these notices could be sent throughout the network by effectively being given priority over all other traffic, then propagation delays could be held in the range of tens of milliseconds for notice propagation, much excess traffic on the network could be avoided, and the simulation software dealing with allocation of contention channels would avoid the need to deal with deinterleaving complex sequences of packets.

There are obviously other possible situations involved in dealing with communications traffic or ECM/ECCM where the need will arise to carefully determine an efficient way to handle technical details of the process which do affect the end-user's view of the process. The SIMNET effort must include a fairly extensive study of the nature of battlefield communications systems, the extent to which the technical details of the system affect their implementation within the simulation, and the means by which efficient handling of the processes can be developed for SIMNET.

5.0 SIMNET MESSAGE PRIORITIZATION

Sections 3 and 4 taken together raise several issues concerning the ability of Ethernet environments to support SIMNET simulations. Even though SIMNET applications carrying voice and data may not have traffic profiles identical to those cited in the research papers above ([4] and [5]), the voice/data results of [5] serve to raise the point that a standard Ethernet configuration which does not favor a voice packet over a data packet may not serve the voice needs of a simulation, while a non-standard implementation favoring voice may begin to create substantial delays in the transmission of data packets. This delay in data packet delivery can then be further aggravated by the fact that not all stations on an Ethernet can obtain the same throughput, with throughput ratios easily reaching four or more. SIMNET by itself is only an information-handling protocol: it cannot control network activities and prioritize traffic. The SIMNET system could conceivably prioritize the order in which received SIMNET packets were acted upon, but such effects would undoubtedly only alter the service to a given packet by a few milliseconds.

If, for example, 100 millisecond data delays were experienced on each of two Ethernets connected by a longhaul network (as shown possible in [5]), then even with no consideration of the gateway and propagation delays of the long-haul segment connecting them, it is clear that updates for high velocity aircraft would become unacceptably large , perhaps in the neighborhood of 0.25 seconds. This is the upper limit which we derived in Section 4.2 for real updates of high velocity aircraft status, and that value was based on fairly conservative assumptions. If the effect of station distribution along the network were also taken into account, as described in Section 3.2, one or both of the terminating Ethernets involved in a transaction could impede the message by as much as 0.4 seconds. Delays of up to a second then begin to look possible. Clearly such delays preclude the simulation of high speed aircraft, and even on one Ethernet, formation flying aircraft might find the real update rate for position orientation data to be unacceptable.

The other major point of the above section was that accommodation of C^3 training clearly entails the transmission of simulated communications: it was shown that the delay and delay variance characteristics of a SIMNET simulation might greatly complicate the handling of simulated communications, especially for contention-accessed channels.

None of the above results represent completely rigorous results drawn from potential SIMNET scenarios. However, the basis of the conclusions above represent traffic profiles drawn from [5] which could be anticipated to be within the reasonable range for some SIMNET scenarios. The results in [4] and [5], which form the crux of the argument given here, were both based on moderate Ethernet loadings, certainly in the range which might be expected in a SIMNET scenario linking remote sites. Further analysis of the possibility that a standard Ethernet implementation might not support some SIMNET scenarios therefore seems germane. Certainly no prioritization scheme relying only on the receiving simulator acting first on high-priority packets can compensate for delays in the indicated range.

As has been suggested previously in this paper, it might well be the case that extending SIMNET to the support of close air support and C^3 activities requires a communications network which supports message prioritization. In many simulations, the proportion of packets representing status updates of high velocity aircraft would be quite small. Also, if voice traffic for common channels were preceded by notice packets, as suggested in Section 4.3, the the network would be relieved of considerable traffic, and the receiving software would be relieved of considerable complexity in the accommodation of voice traffic. The proportion of traffic consisting of such notices would probably be a small proportion of all traffic.

Thus, applying message prioritization to these two types of traffic (high velocity aircraft status and communications notice) would normally involve some small amount of the total traffic, perhaps 10%. If the notice traffic were to be used to preclude the transmission of unnecessary packets (i.e., packets representing losing contenders for common channels), then the net traffic load offered by SIMNET might actually go down. This would almost certainly be the case if much of the traffic precluded by notice packets were voice traffic.

Ethernet is a CSMA/CD system, and as such, access to the channel is gained without any central control, and, likewise, no control overhead. The price paid, however, is the relatively low saturation level (about 32% of channel capacity, as shown in [4]). Is there any benefit to be gained by somehow prioritizing traffic, and can that benefit be realized without major modifications to the Ethernet protocol? F.A. Tobagi addresses this question in a paper entitled "Carrier Sense Multiple Access with Message-Based Priority Functions" (see [8]). In this paper, he outlines a means of providing message-based priority to an Ethernet in which the individual messages within any priority class remain in CSMA/CD contention with each other, but higher priority traffic always gains access to the channel before lower priority traffic. The priority protocol can be made preemptive (i.e, higher priority traffic interrupts lower priority traffic already in transmission), or non-preemptive.

The essence of Tobagi's scheme is as follows. At the end of any message transmission, an interval begins which contains a slot representing each message priority in the system. The slots are temporally arranged with highest priority first, and each slot is long enough so that there is an allowance for timing skew (relative to the end of the last transmission) up and down the bus. If any station has highest priority traffic to transmit, that station puts a burst of carrier in the first slot. All other stations sense that burst of carrier, and only stations with equal priority traffic may then contend for the next access to the channel (which begins immediately after the priority slot). If no station informs all other stations of priority traffic in the first slot, then all stations with the next lower priority of traffic may place a carrier burst in the second slot, *etc.*

In effect, this is a message prioritization scheme which does not call for centralized control, and the only overhead for which is the priority slots following each packet transmission on the Ethernet. In general, the aggregation of these priority slots constitute a very much shorter interval than an ordinary packet length, so the resulting overhead is very low. Of course, all packets of the same priority still compete by CSMA/CD, so the fundamental nature of the Ethernet access protocol does not change, and the overall efficiency of the system does not increase.

However, by prioritization, some class of traffic can gain preferential treatment. If that traffic is a fairly small percentage of the total, then the remaining traffic will not be appreciably affected. In this way, the potential difficulties for some types of SIMNET traffic could be ameliorated. The results shown in [8] were derived by dividing traffic into two priority classes. Two "slots" of channel time were allocated to the prioritization scheme, and the numbers of high and low priority messages were equal, but the length of high priority messages was one-tenth the length of low priority traffic. The consequences of the use of the prioritization scheme was that the high priority packets tended to experience roughly one-fifth the delay of the low priority packets, even at high network loadings where packet length becomes a minor part of the delay.

As for the research done in [4] and [5], this work does not contain results directly applicable to an Ethernet application of SIMNET. However, it does reflect the fact that Ethernet can be prioritized at a modest cost in overhead, and that such a prioritization can demonstrate very much better handling of the priority class of traffic. Clearly, the extent to which the lower priorities of traffic suffer is dependent on the relative proportion of overall traffic which receives high priority treatment. In a SIMNET application, it seems probable that the higher priority traffic

could easily be limited to ten percent of the total, which was the ratio used in the work cited above.

Another even simpler prioritization scheme for CSMA/CD is possible, based on the concept of persistence. In this case, each station with a backlog of transmission traffic, upon sensing that the channel is idle, will wait a probabilistically determined length of time before trying to seize the channel. Stations with higher priority traffic wait, on average, a shorter length of time, so high priority traffic tends to get favorable treatment. Of course, in this case, the higher priority traffic may not always get favored treatment, unless the delay for lower priority traffic is set so large that, on average, much channel capacity is wasted in the dead time between messages. Of course, the exact tradeoff between a system such as described in [8] and the simpler persistence scheme described here would depend on analysis of specific scenarios.

Suffice it to say that various relatively simple schemes exist by which to prioritize Ethernet traffic. Additionally, the emerging OSI standard supports message prioritization as part of its Internet Protocol. Thus it is reasonable to address the need for and practicality of message prioritization for SIMNET implementations on a variety of underlying network architectures. In the next section, we will discuss a possible outline of study for this subject.

6.0 SIMNET NETWORK ANALYSIS - A PROPOSED STUDY PLAN

The above sections discuss the timeliness constraints of SIMNET simulations, together with known results concerning Ethernet performance. Although the Ethernet results cited (references [4], [5], and [8]) are not specifically based on SIMNET scenarios, they do reflect behavior of Ethernet operating at medium traffic loads. Extrapolated to SIMNET circumstances, they indicate that an Ethernet carrying voice traffic together with data, and interlinked by a long-haul network to another Ethernet may experience delays between 0.5 and 1.0 seconds. We have demonstrated in Section 4 that these delays may prove unacceptable in the transport of certain data types, such as position/orientation information for high-velocity aircraft, and data representing C^3 traffic in the system. As discussed in the above section, message prioritization is an option possible on Ethernet and in OSI-based networks, and such prioritization might be very useful in alleviating potential delay problems in Ethernet.

It would seem, therefore, a wise investment at this juncture in SIMNET development to undertake a close scrutiny of SIMNET and its proposed supporting network architectures. Such a study would focus on the resolution of the question of whether SIMNET may need to exploit message prioritization as a way to alleviate possible delay problems for certain types of SIMNET traffic. Briefly, such a study would be partitioned into the following tasks.

1. Examine possible SIMNET simulation scenarios, and extract expected traffic loading, in terms of the physical network assets which might be involved, the types of packets generated, and the time constraints on delivery of those packets.

2. Examine the various candidate network architectures which might support SIMNET simulations relative to their suitability and modifiability to support the timeliness constraints of SIMNET traffic.

3. From all of the above, identify the most favorable low-risk approaches for adequate support of SIMNET traffic with message prioritization or other performance enhancement techniques.

4. Determine by analysis and simulation the best techniques from those surviving the above three steps, and then search for the optimum parametrizations of the chosen techniques.

A study such as the above would consume about two to three man years of labor, and could be carried out over a one to two year time frame. Based on the analysis presented in Sections 2 and 3, such a study could possibly contribute greatly to the total utility of SIMNET, especially if the result was that SIMNET could thereby accommodate a substantially enlarged scope of training activities (e.g., close air support and C3 operations) and/or encompass a greater collection of physical networks within a single simulation.

For these reasons, Harris GCSD encourages funding of a network architecture study as described above. Harris GCSD, and more broadly, Harris Electronic Systems Sector (of which GCSD is a part), is a national leader in advanced network design technology. A resume of Harris ESS credentials in network design technology follows.

7.0 HARRIS ESS NETWORK DESIGN EXPERIENCE

Because Harris Electronic Systems Sector (ESS) is a systems design organization, the approach to communications system design at Harris is tinged with a strong tendency toward the practical. ESS personnel recognized some years ago that the small network analysis methods of the past were becoming obsolete, because the network models (i.e., circuit-switched or packet-switched single media, with fixed routing) which were so well-supported by those computational tools were becoming obsolete. As a result, research into new analytical concepts and tools is already well underway at Harris, and very powerful and unique tools have been developed to handle this new class of large complex networks. Several relevant projects will be discussed below.

7.1 The Multimedia Network Design Study (USA AIRMICS)

To date, very little research has been done on the use of multiple media in networks. What has been done has been limited to *ad hoc* contexts. The AIRMICS study is a three-year study funded by the Army, to examine the issue of multi-media network optimization for large-scale strategic communications requirements.

In the first year of this study (now complete), multi-media networks were characterized generically, and steady-state modeling techniques were applied to examine the optimum proportioning of available media types against offered traffic of several types. In particular, the traffic types were characterized by their timeliness, accuracy, and bandwidth requirements, and media were characterized similarly by delay, error, and bandwidth properties. The steady-state model developed provides a means to assess the optimal way in which to apportion the media amongst the traffic types, given the volume of each type of traffic expected, and the availability of the media.

The use of a steady-state model in this study allows rather fast convergence to solutions, but will not provide the resolution that one would obtain by simulation. The benefit of this approach is that very much larger network structures can be practically evaluated, and the true optimum mix found for any set of input scenarios can be determined.

The second year of the study, now underway, will build on the first year's results. In the second year of effort, the steady state model will be used to ascertain the optimum use of network resources while the network is "quiescent" , i.e., not supporting active defensive actions, and not itself under attack. Then the network, using the optimization results as baselines, will be simulated through stressed scenarios to determine network dynamic behavior. This steady-state first, simulation second approach has a large technical advantage over simulation alone, and that is that the use of simulation only must proceed from a heuristic assessment of the network's optimum quiescent configuration, and the simulation results may therefore cluster around a less than optimum quiescent scenario. The result is that the analysis of the stressed network may conclude that certain response mechanisms are most appropriate when they in fact represent optima determined around the wrong quiescent point.

The third year of this study will focus on the important but much neglected relationship of mission-oriented metrics and engineering metrics of communications systems. A modeling technique has been identified (Generalized Activity Networks, or GAN) which can capture very clearly the relationship between mission metrics and engineering parameters, and can allow precise expression of these relationships. This is an area of study not well-explored in the network community up to this point, and the GAN technique should break new ground in providing a well-formulated and rigorous way to relate mission requirements to network

specification.

The subjects to be examined in this study will result in development of a powerful set of tools which will apply directly and uniquely to the design of the large, complex network architectures of the future.

7.2 The Air Defense Initiative Communications System Design Study (USAF RADC)

The overall objective of this Air Defense Initiative (ADI) Study is to develop a surviving and enduring communications system design that will permit the near-term demonstration of an "intelligent" multimedia communications system supporting the surveillance, engagement, and command and control aspects of the air defense mission. The two main thrusts of the study are to

1) develop a network architecture that is compatible with Distributed Sector Operations Center/Survivable Command and Control Center (DSOC/SC^{2}C) operations;

2) design a multi-media Intelligent Communications Controller (ICC) and multimedia network management algorithm which will manage the communications nodes of this system.

The system nodes will be transportable communications and command and control shelters which can assume control of an air defense sector if the fixed command center is destroyed. The system must connect fixed, land-mobile, transportable, and space-based nodes, and must use all available media in order to survive and reconstitute when necessary. Gateway capabilities to non-ADI communications assets, such as JTIDS and DDN will be included in the system design.

To support the transparent use of multiple media, the system nodes will rely on the ICC. The ICC will intelligently manage system resources at both the nodal and network level. The mechanisms implemented by the ICC can be decomposed into four major functional areas -- the transport layer, the network layer, the link layer, and the network manager. The ICC will be designed to conform to the ISO OSI network model, and will perform the following major functions:

1) simultaneously handle both digital data and voice traffic;

2) provide both circuit-switched and packet-switched services;

3) transparently manage the selection of media on a transaction-by-transaction basis;

4) adapt in real-time to changes in topology, equipment availability, traffic loading and mix, and network performance.

Two major aspects of this system design are the initial robustness of connectivity, and the intelligent control of the connectivity as it changes or is degraded. Intelligent control must sense changes in topology, traffic loading and mix, and must do so using relatively efficient mechanisms that do not themselves require massive communications overhead.

Two major networking concepts are being analyzed in conjunction with this study. The first is the "network of networks" concept, which retains single-media networks as integral entities, but achieves high connectivity via gateways between them. The second is the "flat" or

multigraph organization, in which all links of any media type are integrated more uniformly into a single network.

The current analysis, development, and verification of the ICC network structure and control algorithms is being carried out on a network of Sun 3/50 and 3/60 workstations dedicated to this task.

7.3 Distributed Network Vulnerability Analysis (USAF RADC)

The Distributed Network Vulnerability Analysis (DNVA) study was conducted by Harris to determine a means to generically and uniformly lead analysts through a careful evaluation of communications network vulnerabilities. This study, recently completed, has three main parts -

1) creation of a major taxonomy of network function and an analytical methodology to assist the vulnerability analyst in structuring a comprehensive and complete analysis.

2) design and implementation of a data base program which will support and greatly facilitate future DNVA tasks,

3) a full analysis, using the DNVA methodology, of a complex SDI network architecture.

The development and application of the DNVA methodology will greatly improve the network design process in the future with respect to the analysis and preclusion of system vulnerabilities to hostile action. Harris has designed this methodology, and is currently in the unique position of having the expertise to comprehensively apply this technology to large complex network designs. Implementation of this methodology via the computerized data base will make this a standard tool, available in the future to all military agencies involved in network design.

The three efforts cited above represent large technology components of a military multi-media network design. Of course, there are many other components of such a design, and Harris has, through other studies, IR&D's, and programs, developed a pool of expert credibility in all the relevant disciplines. Equally important is that these activities have allowed the development of a comprehensive tool base to support and verify the network design process. These efforts are described below.

7.4 The GENESIS Network Simulation Tool

The GENESIS network simulation tool was developed to allow simulation of very large complex network designs (see reference [9]). This is normally difficult within the contexts of the usual simulation languages, because they require that the system model be "bent" into the fixed paradigm of the simulation language. (I.e., the simulation language implements limited types of transactions involving limited types of data structures, between entities of certain predefined types.) If one chooses not to use a simulation language, then the usual alternative is to create from whole cloth an ad hoc simulation of the specific network of interest. This latter approach offers unlimited flexibility and resolution, but generally requires a substantial investment of programmer effort.

GENESIS provides an approach to simulation that is a hybridization of the two traditional approaches. The programmer of a GENESIS simulation creates with no restrictions the precise functionality of nodes and channels in the context of a high-level language (MODULA-2 or

ADA). Once that task is finished, the specific topology involving these nodes and channels, as well as a range of initialization parameters, can be established without any further expenditure of programming effort. This is done by an end-user, who enters the topology and input parameters through a Topology program. GENESIS is in use on several current Harris programs, and has been slated to be used in several more. All experience with it so far at Harris has been favorable

The GENESIS tool has contributed mightily to several SDI network design studies here at Harris, because it is ideally suited to open-ended research effort on complex networks. Furthermore, simulation at more resolved levels of network function builds very efficiently on simulation already developed at lesser levels of resolution. Also, GENESIS is written to provide very rapid execution compared to simulations written in standard simulation languages, and thus several times as many simulation runs can be obtained for the research effort than would normally be possible. (GENESIS includes a new exceptionally fast scheduling algorithm especially developed for the GENESIS package, and published in the Proceedings of the 1988 Southeastern Simulation Conference). Use of GENESIS in a design effort insures that a minimum of the design funds will be spent on programming, and a maximum possible amount of the resources can be devoted to the analytical discovery process. GENESIS was used in the work cited in [7], and is being used for continuing analysis of the Freedom Space Station communications system.

7.5 The Network Stressed Topology Model

An important aspect of network design centers around the analysis of failure modes in the system. Almost all analytical work in this area has issued from the assumption that separate network assets fail independently. A few papers have addressed dependent failure modes, but in all but one case, these papers assume relatively "non-real-world" failure mechanisms, and one gets the impression that the failure mechanisms were chosen because the problem addressed then admitted of a solution. Harris has approached the problem of dependent asset failures for military networks from a purely practical standpoint: that is, the distribution of dependent failures is related directly to the types of hostile actions that would in fact be expected for a military network. This model provides a practical means of evaluating network performance in the face of hostile action. Further work on that effort was done under the (USAF RADC) DNVA study (described above) to develop a comprehensive computer program implementing the model.

7.6 Survivable Communications IR&D

The main objective of this IR&D is to formulate and evaluate network routing algorithms that implement adaptive routing, congestion control, and network reconfiguration mechanisms. The focus of the research is on distributed, adaptive network control in order to most adequately meet the constraints of military environments. This IR&D has also provided the impetus of the two important network analysis tools described above in 7.4 and 7.5.

The focus of this IR&D for the previous year was on the completion of a packet radio network testbed (PRNT) based on CSMA/CD channel access. This system involves twelve nodes built from microprocessor-based computers and packet modem hardware, which can be interconnected in a rich variety of topologies. Experiments on this system involve actual communicating hardware, driven by software algorithms developed under the IR&D. Multi-media gateway software has been implemented on the testbed, and work is now focussed on internetting and routing algorithms.

Current year effort in this IR&D is centered on the optimization of dynamic queueing and

routing algorithms for multimedia stressed networks.

7.7 The Stressed Communications IR&D

This multiple year IR&D primarily focusses on HF networking algorithms and protocols, terrestrial and spaceborne wideband networks having a large number of links and nodes, and on improved performance of meteor-burst communications, link hardening, and multi-media networks. The most recent primary activity in this IR&D has centered on developing survivable, efficient network architectures based on adaptive network management algorithms. The multi-media issues being addressed in this IR&D center on the development of algorithms for multi-media networks considered in two contexts -- single network with multi-channel, multimedia links, versus multiple networks sharing common nodes, and connected by gateways. Research is focussed on management algorithms for link assignment, topology update, packet/message routing, and flow control.

7.8 SDI BM/C3 Architecture Development Study (USAF RADC)

This study was funded by USAF ESD, and involved the development of several BM/C3 architectures for boost and mid-course phase, based on twelve different sensors/weapons combinations. The aspects of network control addressed in this study included topology reconfiguration, adaptive routing, and system survivability in the face of EW or physical attack. This effort involved extensive modeling and simulation.

7.9 The BM/C3 Network Design (USAF RADC)

This study is a follow-on effort to the study discussed immediately above. The current study (> $1 million award) comprises a number of design and analysis tasks, including system requirements definition, network design, and simulation and verification of the network design. Network design issues being addressed include security, survivability, adaptive routing, congestion control, internetting, protocol development, link access, and system architecture. Many of the network concepts under investigation in this study are generic to large multimedia networks. The GENESIS tool is being used with great success to provide simulation in this study, and the DNVA technology discussed above is being applied to this network.

7.10 Army BM/C3 Study (USA)

This comprehensive 27 month study program addresses the communications networking requirements associated with all four SDI tiers -- boost, early mid-course, late mid-course, and terminal phases. The communications requirements were derived for several detailed system architecture alternatives, and communication network designs are being developed for selected candidates. Timeliness for various message and data types is a crucial issue in SDI, so detailed simulations are being done using the GENESIS simulation tool. Major factors influencing communications in these simulations are jamming, node destruction, LPI, COMSEC, and nuclear weapons effects.

7.11 Adaptive Distributed Network Management System Program (USN)

The Adaptive Distributed Network Management Program (ADNMS) is a USN-funded study that was initiated to investigate difficult network management issues unique to SDI that have not been adequately addressed on other research programs. This is a three year program (now in the second year) with the objectives of algorithm design, test, and refinement. The study has concentrated on network management algorithms for space-based networks. The four major subcategories of network management of greatest concern in the study are link assignment, failure/recovery/topology update, adaptive routing, and flow control/congestion. Substantial results have been derived in the area of adaptive recovery from network perturbations.

7.12 Surviving, Enduring Multimedia Communications System

This special system, developed for a proprietary customer, involves the design, fabrication and fielding of a large, multinode, multimedia, transportable communications system. This is a multiyear, multimillion dollar system supporting voice and data communications and switching systems over multiple media and over links of thousands of miles. The specific media involved are SATCOM, HF, meteor burst, landlines, fiber optics, and line-of-sight microwave. This has established at Harris a pool of expertise related to the practical aspects of moving theoretical and conceptual algorithms into the domain of real hardware.

7.13 Dynamic Communication Resource Allocation Study (DCRAS) (USAF RADC, NATO)

This study, currently underway, is examining the post-2000 NATO communications environment relative to the need for real-time, automated selection and control of communications assets. The study is considering rather global issues, such as what is needed to control such a system, what characteristics will be required of communications equipment in such an automated environment, tradeoffs of increased survivability and capacity against costs and implementation risk, and recommendations for integration of current hardware into such a future system. In addition to analysis directed at the above concerns, the study will identify technology development areas which are on the critical path to such a future system.

7.14 MultiFunction Information Distribution System (MFIDS) (USAF RADC, NATO)

This study, currently in progress, examines the requirements and technology involved in meeting the need for tactical information distribution in the post-2000 NATO environment. The first phase of the study is examining present and projected NATO responsibilities and capabilities in order to derive key requirements for the MFIDS. The second study phase will assess the progress in important information distribution technologies in order to identify practical means by which to meet the information distribution requirements of the post-2000 tactical situation. The final phase of the study will suggest that certain technological areas be the subjects of further study, in order to stimulate a development path toward the MFIDS system.

The above-described efforts demonstrate the extent to which Harris Electronic Systems

Sector is currently involved in state of the art military network design; yet these efforts constitute only a fraction of the current or recent efforts at Harris related to network design and analysis. They serve to indicate that Harris is second to none in depth and breadth of experience in the military network design arena.

Harris is eminently qualified in both the arenas of simulation and network design to examine and resolve any potential implementation difficulties which might arise in the transition of SIMNET from a small-scale prototype demonstration to a fully functional system optimized for the greatest scope and utility to its potential customers.

8.0 REFERENCES

1. Thorpe, Jack A., et al., "The SIMNET Network and Protocols", DARPA Information and Science Technology Office, Report No. 7102, July 1989

2. Thorpe, Jack A., et al., "SIMNET Database Interchange Specification", DARPA Information and Science Technology Office, Report No. 7108, July 1989

3. Cadiz, J., et al. (editors), "SUMMARY REPORT : The First Conference on Standards for the Interoperability of Defense Simulations", Institute for Simulation and Training, 12424 Research Parkway, Orlando, FL 22-23 August 1989

4. Gonsalves, T., Tobagi, F., "On the Performance Effects of Station Locations and Access Protocol Parameters in Ethernet Networks", IEEE Trans. on Communications, Vol. 36, No. 4, pp. 441 - 449, April 1988

5. Nutt, G., Bayer, D., "Performance of CSMA/CD Networks Under Combined Voice and Data Loads", IEEE Trans. on Communications, Vol. 30, No. 1, pp. 6 - 11, January 1982

6. Lazowska, E., Zahorjan, J., *Quantitative System Performance*, Prentice-Hall, Inc, Englewood Cliffs, N.J., 1984

7. Williams, A. et al., "Instabilities in Heavily Loaded FDDI Networks", Proceedings of the 1989 SouthEastern Simulation Conference, (Society for Computer Simulation) Pensacola, Florida, 16-17 October, 1989, pp. 83 - 88 ..

8. Tobagi, F. , "Carrier Sense Multiple Access with Message-Based Priority Functions", IEEE Trans. on Communications, Vol. 30, No. 1, pp. 185 - 200, January, 1982

9. Doner J.R. , "GENESIM", IEEE J. on Selected Areas in Communications, Vol. 6, No. 1, pp. 172 - 179, January 1988

Position paper: A Proposed Format for the Vehicle Appearance PDU

Ray Fitzgerald
Software Engineer
Evans & Sutherland

This paper proposes a format for the vehicle appearance PDU more conducive to a general standard, which is being proposed of SIMNET.

We propose that the vehicle appearance PDU look as follows:

```
VehicleAppearanceVarient sequence {

        -- general information (applies to all vehicles)
        vehicleID               vehicleID (48),
        vehicleClass            vehicleClass (16),
        location                array of three 64 bit geocentric Cartesian coordinates,
        rotation                array of three 32 bit Euler angles,
        timeStamp               unsignedInteger (32),

        -- specific information (based on vehicleClass)
        specific                choice (vehicleClass) of {

            -- A simple moving vehicle, without a turret:
            when (vehicleClassSimple) simple sequence {
            appearance              unsignedInteger (32) - meaning dependent on
                                    vehicleClass,
            engineSpeed             unsignedInteger (16),
            locationVelocity        array of three 32 bit velocities in meters/second
            }

            -- A Tank
            when (vehicleClassTank) tank sequence {
            appearance              unsignedInteger (32) - meaning dependent on
                                    vehicleClass,
            engineSpeed             unsignedInteger (16),
            locationVelocity        array of three 32 bit velocities in meters/second,
            turretAzimuth           angle (32),
            gunElevation            angle (32)
            }

        }
}
```

If our suggestion from our first position paper "On Adopting the SIMNET Local Area Network Protocol as a Local Area Network standard", to only send the non-changing information when those fields are needed (i.e., at initialization and when a new vehicle enters the exercise) is not adopted, then we feel the vehicle appearance PDU should be changed into the following format.

```
VehicleAppearanceVarient sequence {

        -- general information (applies to all vehicles)
        vehicleID              vehicleID (48),
        vehicleClass           vehicleClass (16),
        force,                 forceID (8),
        location               array of three 64 bit geocentric Cartesian coordinates,
        rotation               array of three 32 bit angles,
        timeStamp              unsignedInteger (32),

        -- specific information (based on vehicleClass)
        specific               choice (vehicleClass) of {

            -- A simple moving vehicle, without a turret:
            when (vehicleClassSimple) simple sequence {
            appearance             unsignedInteger (32) - meaning dependent on
                                   vehicleClass,
            guises                 VehicleGuises - two 32 bit object types,
            markings               VehicleMarkings - character set (8) and 11 char (8),
            capabilities           vehicleCapabilities (32),
            engineSpeed            unsignedInteger (16),
            locationVelocity       array of three 32 bit velocities in meters/second
            }

            -- A Tank
            when (vehicleClassTank) tank sequence {
            appearance             unsignedInteger (32)-meaning dependent on
                                   vehicleClass,
            guises                 VehicleGuises - two 32 bit object types,
            markings               VehicleMarkings - character set (8) and 11 char (8),
            capabilities           vehicleCapabilities (32),
            engineSpeed            unsignedInteger (16),
            locationVelocity       array of three 32 bit velocities in meters/second,
            turretAzimuth          angle (32),
            gunElevation           angle (32)
            }

        }

}
```

The following paragraphs will summarize our reasoning for suggesting such a format. We will primarily investigate the reasons for changing the original format layed out in the July 31, 1989 "The SIMNET Network and Protocols" manual. We will begin by revisiting the 'angles vs rotation matrixes" issue we looked at in our last position paper. In that position paper we stated that two vital abilities (time correction and extrapolation/dead reckoning of heading, pitch and roll) were lost by sending a rotation matrix. We would like to restate this as these two abilities would not be practical in a typical* CIG, if a rotation matrix were sent. Since it is possible to do time correction through the creation of another matrix and the subsequent matrix multiply of the two. Dead reckoning could also be done by sending an incremental rotation matrix in the PDU and multiplying it against the positional rotation matrix for incremental updates. The reason we contend that time correction and dead-reckoning would no longer be practical in a typical CIG, is that the typical CIG front end does not deal with matrixes at all. In general CIGs time correct and extrapolate/dead reckon with Euler angles and pass these updated angles to specially built hardware, which creates the rotation matrix and uses it accordingly. For most CIGs to work with a rotation matrix coming over a network, the host would have to change the rotation matrix back into angles and send them to the CIG as angles to be manipulated. The McDonnell Douglas experience with SIMNET and the Paragon CIG is an excellent example of the difficulties added by sending a rotation matrix.

Another point which has not been raised as yet is the ability to do simple extrapolation of heading, pitch and roll, from previous positional data, without the use of a velocity. This is another common practice which would be impractical with a rotation matrix.

The ambiguity of Euler angles, was mentioned in the Jan 1990, NSIA conference as a reason to send a rotation matrix. This point is obviously not valid, since these ambiguities have been solved by airplane hosts for years. Another point raised in the conference was that of defining a different vehicle class for sending angles and leaving the tank vehicle class with the rotation matrix. This point is also not valid since any particular CIG would still have to be able to deal with a rotation matrix if a tank were within its field of view.

In short, we feel that sending a rotation matrix to a CIG that was built to deal with matrixes in the front end, is probably efficient and practical, but since most CIGs are not capable of practically dealing with matrixes and SIMNET is seeking to become a standard, we believe it should be able to be used practically by off the shelf CIGs.

The second proposal from our first position paper, to only send non-dynamic data when it is necessary was also discussed at the conference. Three reasons were given for leaving these fields in the vehicle appearance PDU: 1) An increase in the size of a packet did not noticeable affect the throughput (number of packets delivered per time period) in an Ethernet environment. 2) Sending the non-dynamic data separately would add a level of complexity to the protocol. 3) The limits of the network are not even being approached and higher performance networks are on the horizon. Reason one is not acceptable, since in a token ring network the throughput (number of packets delivered per time period) is directed related to the size of the packet. Reason two is also mute, since the MCC will most likely be sending information about the state of the database to activated vehicles, and could easily add the non-dynamic data about each active vehicle to this information. Reason three is an interesting point but does not give any reasoning for making packets larger than they need to be. We believe the ultimate limitation of SIMNET will be in network throughput.

*every CIG known to the author except for BBN's.

We will now look at the changes proposed for the general format of the vehicle appearance PDU. We propose that only those fields which are general enough to apply to any vehicle should be incorporated into every vehicle appearance PDU. We contend that only the following fields meet this criteria.

vehicleID	- some mechanism must be available to identify a vehicle
vehicleClass	- needed to relate the format of the PDU
location	- every vehicle's origin must be located somewhere
rotation	- every vehicle must have an orientation
timeStamp	- needed for time correction
force**	- every vehicle is either on one side or the other

We believe the following fields do not meet the criteria for the following reasons:

guises**	- not all vehicle classes will have guises.
appearance	- meaning should be dependent on the vehicle class (32 bits is not enough bits to cover all appearances of all types of vehicle classes), and some vehicles might only have one appearance.
markings**	- not all vehicles will have markings (Dismounted Infantry, camouflaged vehicle, stealth vehicle).
capabilities**	- meaning should be dependent on the vehicle class (32 bits is not enough bits to cover all capabilities of all types of vehicle classes), and some vehicles might only have one capability.
engineSpeed**	- not all vehicles will have an engine (DI), some might have two (F16).

The benefits of this type of format are that only applicable fields to a vehicle class are sent and more flexibility in the sizes and meanings of the fields under each vehicle class is available. For example, if a certain vehicle class has more than 32 abilities/appearances, this format could easily accommodate the vehicle, where the other format would be trying to put the abilities/appearances of all types of vehicles into just 32 bits. The marking field is another field where flexibility will be very beneficial. Some vehicle classes might need 20 characters, and 2 symbols not in the ascii character set (e.g., markings on the sides of aircrafts). The current format could not handle such a case.

In summary, we feel this format is more practical to the typical CIG, uses the network more efficiently, and handles the expanding needs of SIMNET more gracefully than the current format of the vehicle appearance PDU.

**non-dynamic field

Data Representation Issues
Steve Seidensticker
Logicon, Inc.

There are five areas of basic data representation that must be addressed in the Defense Simulation Interoperability Standard (DSIS). They are FloatingPoint Format, String Variable Format, Byte Ordering, EnumerationRepresentation, and Angle Representation. Each is treated separately below:

Floating Point Format

The IEEE has established a floating point format for representing realnumbers. This format has been adopting by virtually all computer designers. It has become the standard for representing real numbers in all computerinstruction sets developed in the last ten years. This includes all themicroprocessors that have been developed in that time (e.g. Motorola 68000 series, Intel 80x86 series). This format has also been adopted by all the RISC (Reduced Instruction Set Computers) designs. However, there are a number of older instruction sets still being used by the traditional minicomputer vendors (Digital, Encore, Concurrent) in current products that use proprietary floating point formats. The use of these computers in simulators adhering to this standard will require the translation of variables in these floating point formats to the format identified by this standard.

Recommendation:

That the DSIS adopt the IEEE floating point data representation standard as the standard for representing all real values flowing between simulators.

String Variable Format

There are three basic considerations for the standardization of string representation.

The first is the representation of the characters themselves. The American Standard Code for Information Interchange (ASCII) has been universally accepted by the computer industry as the means for representing alphanumeric characters within computers. There are virtually no other means in current use for the DSIS to consider. But, for the sake of completeness, the DSIS should include a statement adopting the ASCII standard for alphanumeric character representation.

The second consideration in representing strings is the order in which the characters appear in the message. In modern computer designs, in which each byte of memory has a separate address, this

is not a problem. However, many older systems use a 16-bit or larger word as the basic addressed element. This gives the software components of the system several choices in how ASCII characters are packed into a word. Some companies have chosen to put the first character of a sequence into the lower order portion of the word and other companies have chosen the opposite method. The result is that when a string variable is transmitted as part of an intersimulator message the character sequence may not be what is expected by the receiver. That is, the string "ABCD" may appear in the message as "BADC".

The third consideration in representing strings is the method of determining when the string ends. There are two popular methods for this. The first uses the first byte of the string variable itself to hold a count of the number of characters in the remainder of the string. The advantage of this scheme is that the length of the string is readily available. The disadvantage is that the maximum length of a string is limited to 256 (the maximum value that can be represented in a single 8-bit byte). The other popular method uses a unique value (non-ASCII) to mark the end of the string. This has the advantage of defining a string variable of virtually any length. The disadvantage is that processing software must search the entire string for the special end-of-string marker to determine its length. Neither of these methods has found universal favor. The HAVE MODULE(formerly MODSIM) program has dealt with this issue and has defined astandard regarding it.

Recommendation:

That the DSIS adopt the standard(s) established by the HAVE MODULEprogram for the representation of strings in intersimulator messages.

Byte Ordering

The order in which bytes are transferred between the main memory of acomputer and its CPU registers differs between computer manufacturers. One order, generally referred to as "Big Endian", puts the lowest addressedbyte of a series of bytes that make up a single variable in the mostsignificant position in a CPU register. In a "Little Endian" system the lowest addressed byte occupies the least significant portion of the CPU register. This is a hardware issue and must be resolved before a Big Endian system can exchange information with a Little Endian system in real time. Motorola, a dominant company in current CPU designs, uses the Big Endian mechanism. Intel uses the Little Endian method. Both SIMNET and the HAVEMODULE programs have adopted the Big Endian standard due to the fact that both use Motorola CPUs in their computers.

Recommendation:

That the DSIS adopt the Big Endian (Motorola) standard for internal

byte ordering.

Enumeration Value Representation

Several modern programming languages (including Ada) make extensive useof enumeration variables. For example, one can define a variable called Color and the values it might hold can include yellow, green, blue, etc. The way that the values are actually stored and transmitted in messages are via a series of numeric values with one value representing each enumerated value. These values are assigned generally in the order in which the enumerated values are listed when the variable is defined (e.g. yellow=1,green=2, blue=3, etc.) Differences occur in the way the list is normalized(e.g. does is start with zero or one) and how many bytes each value occupies. A 1-byte enumeration variable can only have 256 values. The HAVE MODULE program has dealt with this issue.

Recommendation:

That the DSIS adopt enumeration value representation standard(s)established by the HAVE MODULE program.

Angle Representation

Several papers at the DSIS January 90 conference proposed the use of floating point variable to hold angular values. One paper (Robert Glasgow) proposed an integer method generally referred to as BAMs (Binary AngleMeasurement). The floating point format is straightforward and is readily understood by most programmers. BAMs are not widely used and require an hour or so of instruction on their use. The difference is in the efficiency of resources used by the two methods. A 32-bit floating point value representing latitude and longitude is not accurate enough to express position in a DSIS scenario. A 64-bit floating point variable is overkill. Latitude and longitude expressed in a 32-bit BAM, however, will permit expression of position at the equator to about one centimeter. The difference in the efficiency is due to the fact that not all bits in a floating point number are always available to represent values.

Angle values are very prevalent in modern vehicle simulation (e.g. lat/long,pitch, roll, yaw, pitch/roll/yaw rates, pitch/roll/yaw accelerations, controlsurface deflection, gun elevation/azimuth, etc.) . Their efficient transmission and processing will have major impact on communication bandwidth and computer resources required.

Recommendation:

That the DSIS program undertake an in-depth analysis of the differentmeans of expressing and processing angular values before establishing astandard representation of them.

The Standardization of Protocol Data Units for the Interoperability of Defense Simulations

A Position Paper by:

Christina L. Pinon
Research Associate
Institute for Simulation and Training

The Standardization of Protocol Data Units
for Interoperability of Defense Simulations

INTRODUCTION

The SIMNET program has been a pioneer in "multi-interconnected-simulator" training for the army. As the benefits of this training have been realized a recognition of the need for more training of this type has become more prevalent in the past few years. In an attempt to answer this need, a standardization process has begun in order to keep the development of these trainers in line with the growth in technology and to allow greater interoperability of simulations.

STANDARDS IN PROCESS

Standardization of networking protocols has been going on for nearly 15 years and much progress has been made in that time. One of the most well known results of this process is the Open Systems Interconnect Reference Model (OSI) developed by the International Organization for Standardization (ISO). This paper will make use of this model in order to discuss protocol standards. The model consists of seven layers, each with its own function for communication between two applications. Since this model is well documented (see [1],[2] and others) the specifics will not be repeated in this paper. In spite of the time spent on the standardization process, many different protocols exist. The ISO recommendations are not universally accepted, as the number of different computer architectures that are in existence would point out.

Therefore, to assume that a standard protocol suite** can be defined for interoperability of defense simulations, within the next few months, is assuming far too much. However, serious consideration needs to be made toward that end in the near future. It has been recognized that simulation needs have been identified to a point where a protocol standard can be developed for the highest layer (or layer 7 - the application layer, of the OSI model). Standardization of other layers would proceed from there.

**A protocol suite is a group of protocols where each protocol represents a particular layer of the computer architecture.

OUR MISSION

The current mission at the Institute for Simulation and Training (IST) is to develop a standard for Protocol Data Units (PDU) on the application layer only. This is the standard that will be presented in late May this year. The Simulation PDU's of the

SIMNET protocol will be considered a baseline for this effort. Other PDU's (Association PDU, Data Collection PDU) will be examined as well. In addition to the SIMNET protocol, position papers presented at the January conference and papers received by IST since then will also be considered.

FUTURE WORK

It is clear that there are many more issues concerning interoperability that need to be addressed in the near future. Issues concerning network architecture, terrain database, timestamping, and semi-automated forces are a few of the topics being considered for future work at IST. Eventually decisions will need to be made concerning needs for standards in these areas as well. Meanwhile, development of application layer protocol will proceed with an eye upon these other issues.

REFERENCES

[1] Tannenbaum, A. Computer Networks, 2nd Ed. , Prentice Hall, Englewood Cliffs: 1988.

[2] Stallings, W. Handbook of Computer Communications Standards The Open Systems Interconnect (OSI) Model and OSI-Related Standards, Howard W. Sams & Co., Indianapolis: 1989.

Transport Layer Protocol Options for the Distributed Simulators Architecture (DSA)

L. Michael Sabo
Martin Marietta Team
SSDS, Inc.
1101 W. Mineral Ave.
Suite 200
Littleton, CO 80120

Transport Layer Protocol Options for the Distributed Simulators Architecture (DSA)

February 13, 1990
L. Michael Sabo
Martin Marietta Team
SSDS, Inc.1101 W. Mineral Ave.
Suite 200
Littleton, CO 80120
(303)798-5520

1.0 Executive Summary

Within the interactive simulation community debates are occurring as to what impact the Distributed Simulators Architecture (DSA) [1] and its suite of open protocols will have on the overall performance of simulator to simulator communications. This position paper will address those concerns as well as describe transport layer options for DSA. A transport layer communications service provides an end-to-end logical connection between devices, independent of the underlying physical network. Essentially what this means for interactive simulation is that a standard transport mechanism will provide an end-to-end data path between simulators, regardless of the physical network interconnecting the simulators. Thus, a local area network (LAN), such as Ethernet, or a wide area network (WAN), such as a mesh of 56 Kbps links, will use the same protocol suite down to the network level. To be useful in DSA, and hence applicable for the interoperability of defense simulators, the selected transport protocol must support a high performance multicasting datagram service, not common to all transport protocols. For example, most transport protocols provide a one-toone reliable data stream between interconnected network entities which creates excessive processing and connection establishment overhead. Broadcast communications schemes are not appropriate either since all simulators on a network are required to participate in the same exercise. As a result, a multicast communications service, which provides a one-to-many transmission capability yet will also support multiple simulation exercises on a single network is required. A connectionless datagram service is also necessary because when a simulator transmits a PDU it does not expect an acknowledgement response from each of the receiving simulators, significantly reducing the traffic on the network. This is considered an unreliable service but is used to provide the necessary performance.

This paper will present three alternative transport layer protocols which could be incorporated into DSA. The actual selection of the transport layer protocol should only be made after thorough experimentation and testing.

2.0 Introduction

The need for high performance multicast transport layer services to support distributed systems is not unique to networking defense simulators. This is advantageous since many years of extensive research are directly applicable to the needs of the interactive simulation community. This paper presents the requirements for a suitable transport mechanism and introduces several transport protocols which are viable alternatives.

3.0 Overview of Transport Layer Services

The transport layer provides an interface between the higher layer end-to-end services and the (inter)network environment below. The transport layer provides a transition from the implementation specific subnetwork communications to process-to- process communications which will occur among interconnected simulators. The transport layer is a fundamental building block for (inter)networking interactive simulators.

Many popular standard transport protocols exist, but few address the needs of DSA. The following section details these specific needs.

4.0 Transport Protocol Requirements

To be a viable transport mechanism within DSA, the protocol must be high-performance. This is extremely important. However, a number of additional goals must be met as well. The transport layer protocol of DSA must be capable of multicast, so that selected groups of simulators can interact and not interfere with various other interacting simulators on the same network. In addition, the transport mechanism must provide datagram service. A datagram service is an unreliable transport scheme where messages are transmitted and no response is expected from the receiver as to whether the message was received without error. In interactive simulation, through the use of dead reckoning, some amount of message loss can be tolerated; however, if the number of lost messages or messages received in error become extensive, the simulation can be severely hampered. In addition, a LANs data link control (OSI layer 2) mechanism provides extensive error control which significantly reduces the probability of a transport layer error.

5.0 Protocol Performance

Their are those in the interactive simulation community who feel that DSA, and its suite of open protocols, will have a large performance impact on simulator to simulator communications. Much of this concern centers on the size of the application layer protocol data units (PDUs). As it turns out, the length of the PDU, which includes protocol headers and user information, is not

the limiting performance criteria of a network unless the network is at or very near saturation. In addition, protocol processing of a transport service is not the major source of processing overhead either [2]. The actual source of the performance overhead is the host processors operating system. In order to process an incoming packet, the operating system must accept an interrupt, allocate buffer space, free the buffer space, restart the I/O device, wake up the processes which handle the packet processing, and reset a timer. The major overhead in processing the packet is that associated with actually manipulating the octets. This would include calculating checksums, separating the protocol header from the data, and copying the information from the I/O device to internal memory. For TCP, the typical operating system overhead is seven times higher than the protocol processing. As this suggests, unless the host operating system handling the incoming data is optimized for this activity, it doesn't matter what protocols are selected; the overall communications performance will not be optimum.

6.0 Alternative Transport Protocols for Consideration

It is extremely beneficial to select an existing transport protocol rather than creating one specifically for DSA. The reasons for this are numerous. Widely available transport protocols have been tuned to increase their performance. Also, when a protocol enjoys a wide installation base it translates into lower cost for equipment, and there is no need to form a standardization community for a unique transport layer. Finally, as technological advancements occur in transport layer services, these could be easily incorporated into DSA.The following paragraphs lists three specific transport protocols which warrant further investigation.

6.1 Versatile Message Transaction Protocol (VMTP)

VMTP, developed at Stanford University by David Cheriton, was originally designed to support remote procedure call (RPC) and general transaction-oriented communications [3] [4]. In addition, VMTP provides distributed real-time control with prioritized message handling, including datagrams, multicast, and security. Another advantage of VMTP is naming of transport-level endpoints. This facilitates process migration, mobile hosts and multi-homed hosts. VMTP has currently been implemented in the Internet protocol stack but it's specification has been forwarded to ANSI for consideration as an international standard. Figure 1 depicts how VMTP would integrate into DSA.

6.2 Xpress Transfer Protocol (XTP)

XTP was designed as a high-performance transport mechanism to be used on the new high speed LANs such as FDDI, 16-Mbits/s token ring, and broadband ISDN [5]. Benchmarks conducted on XTP at the University of Virginia indicate that in requires five to ten times less processing power to attain a given transmission speed that protocols such as TCP/IP, Transport Class 4 (TP4), or Xerox's XNS.

This performance gain can be attributed to the fact that XTP is implemented on a chip rather than in host-based software. In addition, XTP will support a reliable datagram multicast scheme.

XTP has been officially proposed to ANSI committee X3S3 for inclusion as an OSI standard at layers three and four of the OSI model. Figure 2 depicts how XTP would be incorporated into DSA. Note that Ethernet is not listed an optional LAN protocol. The creators of XTP have recommend to ANSI that XTP only be used over high-speed LANs such as FDDI or 16 Mbp/s Token Ring.

6.3 IP Multicasting

IP multica ting has its roots in the Internet community, which is the same group who developed TCP/IP. IP exists at or above the network layer but it is not a transport layer protocol. It is often referred to as layer three and a half. Thus it is recommend that IP be considered in combination with the transport layer services of User Datagram Protocol (UDP) [6]. IP multicasting is defined as the transmission of an IP datagram to a host group. A host group is defined as zero or more hosts identified by a single IP destination address. A host may join or leave any particular host group at any time.

Internetwork forwarding of IP multicast datagrams, very important for distributed simulation over long haul communications facilities, is handled via multicast routers. A host transmits IP multicast datagrams as a local network multicast (much like the SIMNET protocols currently do). If the datagram has an IP time-to-live greater than 1, the multicast router(s), attached to the local network, take responsibility for forwarding it towards all other networks that have members of the same destination group. Those other member networks which are reachable within the IP time-tolive have an attached multicast router which completes delivery by transmitting the datagram as a local multicast. Figure 3 depicts how IP multicast would be integrated into DSA.

7.0 Transport Protocol Working Group

The charter of the Transport Protocol working group is to achieve a consensus as to the transport mechanism for DSA. The transport protocol would be selected based upon research and experimentation. The selected transport protocol would be documented in a report and forwarded to the DSA executive committee for review and comment.

8.0 Conclusions

An adequate transport layer service is required for open interactive distributed simulations. Much research and experimentation is required before a transport protocol can be selected for DSA. Three viable options have been explained in this paper. It should be remembered that work can progress on defining the transport layer protocol concurrently with defining the application protocol for DSA.

9.0 References

[1] Sabo, L. Michael, "Distributed Simulator Architecture (DSA)": Position Paper for the Second Workshop on Standards for Interoperability of Defense Simulators, January 26, 1990

[2] D. Clark, et al, "An Analysis of TCP Processing Overhead", IEEE Communications Magazine, Vol 27 No 6, pp. 23-29, June 1989

[3] D. Cheriton and C. Williamson, "VMTP as the Transport Layer for High-Performance Distributed Systems", IEEE Communications Magazine, Vol 27 No 6, pp. 37-44, June 1989

[4] D. Cheriton, "VMTP: Versatile Message Transaction Protocol Protocol Specification", RFC 1045, SRI Network Information Center, February 1988

[5] E. Hindin, "New LAN Protocol Promises Multi-megabit/s Throughput", Data Communications, pp. 49-54, March 1989[

6] S. Deering, "Host Extensions for IP Multicasting", RFC 1112, SRI Network Information Center, August 1989

Protocol Layering Implications on the Standardized PDUs for Interoperable Simulation

Tom Gehl
IBM
9500 Godwin Drive
Manassas, VA 22110

Protocol Layering Implications on the Standardized PDUs for Interoperable Simulation

INTRODUCTION

The objective of the Workshop on Standards for the Interoperability of Defense Simulations is to determine a standard PDU format which is verstatile enough to interconnect the variety of DoD simulators for tactical training applications. The standards committee has selected the DARPA funded SIMNET PDU format as a baseline. The SIMNET protocol currently is being used in the SIMNET program which encompasses the architecture and development of an interoperable simulation system. For the SIMNET program, BBN designed a network architecture of the SIMNET protocol laying directly on top of the Ethernet protocol. It should be noted that the SIMNET PDUs NOT the SIMNET network architecture is the baseline for the standardization process. In determining the standard, we should concentrate on the PDU format and not try to specify any architectures.

CURRENT STANDARDIZATION PROCESS

In specifying the PDU format, the Interface subcommittee should define a standard which exists as an application layer in the seven layer ISO OSI model. Because of the uniqueness of DoD's requirements for real time distributed training, there is a need for a standard PDU format for the application of tactical training, which currently does not exist. Using the ISO OSI reference model, the application layer is the highest layer protocol in the seven layer scheme. Below the application layer exists the presentation, session, transport, network, data link, and physical layers. Each layer provides a function that the upper layers can use in order to communicate information across the network. Between each pair of adjacent layers there is an interface which defines services that the lower layer offers the upper layers. The set of layers and protocols is called the protocol suite. As mentioned, our purpose is to determine the standard PDU format NOT to determine a standard protocol suite by which all future trainer systems will be specified.

As defined by the ISO OSI reference model, the content of the application layer is up to the individual user. The standard PDU format is needed in order that users on a distributed training system can communicate. Since we are only determining an application layer standard, the standard PDU format should have no field or requirement dependencies on existing protocols in the ISO OSI protocol suite. It is up to the designers of the respective training systems to determine the network architecture. As noted in the committee discussions, latency through the ISO OSI layers can

be a detriment to the performance of a real time distributed training system. But determining which layers are needed when trading between services and latency is up to the designers of the particular system. For instance, BBN chose to design a system which has the application layer protocol interfacing directly to the Ethernet protocol. Sometime in the future the standards committee may have to work on a standard protocol suite to satisfy real time training. But that is up to the committee members, and is beyond the scope of the current standards process.

FUTURE CONSIDERATIONS

Even though we are determining the application layer and not the network architecture, some significant consideration should be noted on the possible effects of layering when interconnection between multiple networked training systems is required. For a distributed training system, the main objective is to communicate the data as quickly as possible in order to minimize latency effects on the training. Thus, the network designs will consist of the minimum amount of layers in order to communicate the data. But some layers provide communication services which could be very useful. For instance, protocols in the network layer provide routing capability which could be useful in filtering information across the gateways. Also, when trying to interconnect different distributed training systems, communication between different network architectures will be an important design issue. Possibly, after the standard PDU formats have been completed, the DoD would desire a standardized network architecture as in the Navy/Industry SAFENET standardization process, which has a full seven layer ISO protocol suite and a parallel lightweight protocol suite.

To allow two distributed training systems to communicate with incompatible protocols, intelligent gateways are required. The function of the gateway is to convert packets from one protocol to another; however, the conversion process can be time consuming and is a potentially significant factor in message latency. There are various designs of gateways just like there are various designs in network architectures. The complexity of the gateway is dependent on the network protocols which the gateway is specified to interconnet. By specifying a standardized protocol suite, the DoD would have some control over the complexity of the interoperable training systems.

As mentioned, interfaces define services that the lower layers offer the upper layers. The interfaces act as access points between adjacent layers. For example, a Media Access Control sub-layer(MAC) provides the interface between the Logical Link sub-layer and the physical layer. The physical layer provides actual communications, and the upper layers

provide virtual communications between the machines. During transmission, the upper layer adds an header containing control information used by a protocol at each respective layer. At the receiving node, the headers are stripped off as the data is moved up the layers. The headers for the layers below do not reach the upper layer. The headers contain service access points which provide a link between the lower and upper layer. This method usually means that like layers of a protocol suite on one node communicate with like layers of the same protocol suite on another node. An example of how messages are communicated between the layers is shown below.

```
                                         ---
Application                              |M|
                                       ------     6/7 interface
Presentation                          |H7|M|
                                    ---------     5/6 interface
Session                            |H6|H7|M|
                                 ------------     4/5 interface
Transport                       |H5|H6|H7|M|
                              ---------------     3/4 interface
Network                      |H4|H5|H6|H7|M|
                           ------------------     2/3 interface
Data Link                 |H3|H4|H5|H6|H7|M|
                        ---------------------     1/2 interface
Physical                |H2|H3|H4|H5|H6|H7|M|
                        ---------------------
                                  |  Physical Medium
                                  --------------------->
```

ISO OSI Layered Architecture

Currently, SIMNET interfaces directly to the Ethernet frame format which consists of a header, data, and trailer, as shown below. The header is a media access point to the physical layer.

```
<------------ Header ---------------><--Data---><-Trailer->
-----------------------------------------------------------
64 bits | 48 bits  | 48 bits | 16 bits | variable | 32 bits
-----------------------------------------------------------
Preamble Destination Source      Type       Data       CRC
         Address     Address
```

Ethernet Frame Format

SIMNET does not make use of any other standard protocols, since the Appearance PDUs lie directly on top of the Ethernet frame format.

The IEEE Project 802 has been working actively on the development of a Local Network Standard. While working in the bottom two layers - Physical and Data Link, Project 802 has divided the Data Link Layer in to the Logical Link Control Sublayer, responsible for the usual link control and logical connection, and below it, the Media Access Sublayer, concerned with a station's physical access to the link. As noted, SIMNET only uses the Media Access Sublayer. Logical Link Control provides a uniform Data Link service to the next layer, so that the upper layer is not affected by the distinctions among the different LAN types. For example, the 802.2 Logical Link Control layer above IEEE 802.3 uses a concept known as Link Service Access Point (LSAP) which uses a 3-byte header:

```
----------------------------
|DSAP    | SSAP    | Control |        IEEE 802.2 LSAP
----------------------------
```

Due to the growing number of applications using IEEE 802 as lower layers, an extension was made to the IEEE 802.2 protocol in the form of the Sub-Network Access Protocol (SNAP). It is an extension to the LSAP header above, and its use is indicated by the value 170 in both the SSAP and DSAP fields of the LSAP frame above.

```
----------------------
 3 bytes  | 2 bytes  |        IEEE 802.2 SNAP
----------------------
protocol ID   EtherType
or org. code
```

Most LANs in the future will have a LLC header included by the LAN adaptor interface card. Also, CCITT, ISO, ANSI, and IEEE are studying a proposal of adding the IEEE Project Standard to X.25, which would provide a homogeneous gateway solution to the problem of linking and interfacing LANs with WANs.

Even though other LANs require LLC headers to communicate to the physical medium, there are commercial bridges which interface the MAC header to the LLC header for different LANs. For example, IBM has a Local Area Network Bridge (8209) to perform a Token-Ring to Ethernet conversion and vice versa. The Token Ring contains an LLC header while the Ethernet does not contain the 802.2 LLC header. In this conversion, the routing information (RI) and the destination service access point (DSAP), source service access point (SSAP), control (CONT), and protocol ID contained in the subnetwork access protocol (SNAP) header are extracted from the token-ring frame and discarded. The destination address (DA), source address (SA), and information field (TYPE and INFO) are copied into an Ethernet frame and sent to the Ethernet LAN. In the conversion from an Ethernet frame to

token-ring frame, the destination address (DA), source address (SA), and information fields (TYPE and INFO) are copied into the respective fields of a token ring frame. The 8209 then retrieves the source routing information associated with the token-ring destination address and inserts these fields and the fixed hex values AA AA 03 00 00 00 (SNAP header) representing the DSAP, SSAP, control and protocol ID fileds into the frame, before sending the frame to the token-ring LAN.

```
                       Token-Ring Frame Format
-----------------------------------------------------------
|SD|AC|FC|DA|SA|RI|DSAP|SSAP|CONT|P_ID|TYPE|INFO|FCS|ED|FS|
-----------------------------------------------------------
              |       Insert/       |
              |       Discard       |
-----------------------------------------------------
|PREAMBLE|DA|SA|                    |TYPE|INFO|FCS|
-----------------------------------------------------
                       Ethernet Frame Format
```

Although the 8209 Local Area Network Bridge performs a conversion between Token-Ring with an LLC sublayer and Ethernet without an LLC sublayer, we are unsure whether there are commercially available bridging products for all other standard networks, present and future. By providing an LLC header requirement in the future standardization process for interoperable simulation, the training application standard protocol can use standards specified by Project 802. Also, the LLC header provides functions which are very useful to the distributed training application. For instance, the receiving simulator could check the LLC Destination Service Access Point to see which application PDU is being sent, and filter frames of information, if the LLC is on the adaptor, before they are accepted by the host processor. This filtering could alleviate processing by host simulators and gateways. Although the effects of the additional latency by the LLC sublayer has not be studied, we believe the latency of the LLC sublayer should be nominal. Thus, the LLC sublayer can provide an important filtering capability, ensure commercially available bridges and gateways to interface future network architectures, and does not detrimentally degrade the performance of the network for distributed training.

Position Paper on
Communicating Change to a Simulated World

P. Wever, Ph.D.
BBN Systems and Technologies Corp.

At the January conference on Standards for the Interoperability of Defense Simulations, discussions regarding the SIMNET Database Interchange Specification (SDIS) [1,2] and the Generic Transformed Data Base (GTDB) format, developed under Project 2851[1], suggested that these two representations are viewed as being in competition as database interchange standards. Rather than frame the issue as a choice between an SDIS format and a GTDB format, this paper proposes to first look at the needs for exchanging terrain information that are likely to arise in distributed simulations and then describe what an interchange standard could do to help meet those needs. A suitable interchange standard might combine aspects of both the SDIS and GTDB approaches.

Distributed simulations will present a particular challenge in communicating changes after the terrain database has been released, perhaps in SDIS or GTDB formats, and implemented on individual simulators. Two types of changes can then occur in the field.

The first type of change is the incremental changes, or updates, that arise when the simulated world is modified by adding features and attributes or by correcting anomalies. These changes take place offline prior to an actual simulation exercise. The more widely a terrain database is used in various training activities the more likely it is than incremental changes will be necessary to support evolving training objectives.

The second type of change is the dynamic changes that occur during a simulation exercise when an individual simulation module induces a change in the simulated world, for example, through some type of weapons effect or combat engineering operation. Although dynamic changes are temporary (for the duration of the exercise) they must be communicated across the network in a coherent and efficient manner. The ability to realistically model dynamic changes to the terrain is particularly important in simulations involving ground fighting units. Managing dynamic change within a simulation

---

[1] Project 2851 Program Office, the Deputy for Training Systems, Aeronautical Systems Division (ASD/YW), Wright Patterson AFB, Dayton Ohio.

exercise is analogous to concurrency control for optimal query and updating in a distributed database.

The difficulty in communicating change in distributed simulations arises because the players have their own "views" of the simulated world. These views are the internal representations of the world that reside in the specific simulation application databases (e.g. for tanks, aircraft, semi-automated forces, etc.). These separate views must be correlated to a sufficient degree that the participants believe they are interacting in the same world. When changes take place, as with any distributed database system, the distributed simulation environment must provide a mechanism for insuring the consistency and completeness of the database. Because the participants can be at widely separated locations (linked, for example, via long-haul networks) the network may be the most efficient way of exchanging this information.

It seems unlikely that coherent change can be managed without a representation of the "objective" simulated world to serve as a reference model. Meaningful change needs to be communicated in terms of (and measured against) the reference model. Incremental updates could be provided for various application databases, for example, in GTDB formats. However, to insure that the application databases are interoperable with one another, any changes that are implemented must be correlated with the reference model. To handle dynamic change each object that can be affected must be capable of being identified so that the change can be communicated unambiguously over the network. The reference model will probably need to have unique identifiers for those potentially changeable objects.

The exchange model developed by P2851 provides for generic application databases (the GTDBs) to be derived from a common data source, the standard simulation database (SSDB). The SSDB appears to comprise potentially all the data pertaining to an area. Without some type of filtering and processing to insure consistency the SSDB will likely not provide a consistent, self-contained model of the objective simulated world. However, a complete high resolution GTDB could be generated to serve as the required reference model. Incremental changes to the GTDBs, together with updates to the reference model, could then be communicated to widely separated simulator sites. In its present form the P2851 exchange model does

not use object identifiers and does not seem to offer any mechanism for supporting dynamic change.

The SIMNET Database Interchange Specification was designed to provide simulation applications with a complete description (the reference model) of the simulated world. The SDIS representation is architecture-independent and defines the contents of current SIMNET databases. SDIS provides identifiers for all objects in the database. These identifiers could be used for communicating changes on an object basis. The SDIS structures are defined using Abstract Syntax Notation One (ASN.1). Basic encoding rules allow ASN.1 specified data to be represented for exchange over communications networks. In fact, it is the potential for using the network as a means of communicating changes to the database (both incrementally and dynamically) that led to the idea of using ASN.1. For similar reasons, the Canadian Government has undertaken a project to develop an ASN.1 based cartographic format [3] for exchanging information over networks.

In summary, managing change in distributed simulations will likely require the use of a reference model to provide a consistent and complete definition of the simulated world. Changes to the world will need to be communicated in terms of the reference model using object identifiers since changes must happen on the "real world" (not inside individual simulators) and communicated at that level. The network itself can provide the means for exchanging this information. A database standard can support coherent change by making the reference model available to the individual simulation applications as the basis for correlating changes. The reference model must also provide the unique object identifiers needed for communicating changes.

REFERENCES

[1] *SIMNET Database Interchange Specification*, Report 7108, BBN Systems and Technologies Corporation, July 1989.

[2] *User's Guide SIMNET Database Interchange Specification*, Addendum to Report No. 7108, BBN Systems and Technologies Corporation, January 1990.

[3] *MACDIF, Specification of the Map And Chart Data Interchange Format*, Version 2, March 1988, Canadian Hydrographic Service.

Position Paper: Byte Ordering for Simulator Internetworking Protocol

Phillip S. Yoo
Digital Equipment Corporation

10-February-1990

Submitted to: University of Central Florida
Institute for Simulation and Training

TIME CRITICAL WORKING GROUP

digital

1.0 EXECUTIVE SUMMARY

Digital proposes that the Simulator Internetworking Protocols adopt Little Endian byte ordering for data. Little Endian representation is compatible with a greater number of machine architectures.

2.0 LITTLE ENDIAN BYTE REPRESENTATION

Bit 31 — 16		15 — 0		
Word 2		Word 0		Longword 0
Byte 3	Byte 2	Byte 1	Byte 0	
Word 6		Word 4		Longword 4
Byte 7	Byte 6	Byte 5	Byte 4	

- Least significant byte is at lowest address
- Word is addressed by byte address of least significant byte .

Architectures compatible with Little Endian:

MIPSCO	R2000, R3000, R60000, etc.
Intel	8086, 80286, 80386, iAPX
National Semiconductor	320000
Advanced Micro Devices	29000
Digital	VAX and PDP-11

3.0 BIG ENDIAN BYTE REPRESENTATION

Bit 31 — 16		15 — 0		
Word 0		Word 2		Longword 0
Byte 0	Byte 1	Byte 2	Byte 3	
Word 4		Word 6		Longword 4
Byte 4	Byte 5	Byte 6	Byte 7	

- Most significant byte is at lowest address
- Word is addressed by byte address of most significant byte

Architectures compatible with Big Endian:

Motorola	680x0, 88000
IBM	370

4.0 RATIONALE

Little Endian Byte Ordering is the appropriate selection for the application protoocol.

1. Byte ordering representation must be established for the application-application data communications.

2. The byte ordering standard should support the broadest number of hardware architectures to accommodate heterogeneous platforms.

3. The principal Big Endian architecture is Motorola. The byte ordering scheme for the Simulation Internetworking protocols should not embrace a single architecture but should allow contractors to a choice of processor architectures.

SIMNET and HIMAD Weapon Systems

Richard P. Gagan
Raytheon Company T3LL21
50 Apple Hill Drive
Tewksbury, MA 01876

Position Paper: SIMNET and HIMAD Weapon Systems

February 15, 1990
Richard P. Gagan
Raytheon Company T3LL21
50 Apple Hill Drive
Tewksbury, MA 01876

PATRIOT and HAWK are the Army's high to medium altitude air defense (HIMAD) weapon systems. Connecting their simulators or training systems to those of other weapon systems via SIMNET or DSIN could support interactions such as coordination with friendly air units for fratricide avoidance; defense of friendly ground forces against ABT and TBM attack; engagement by hostile units during road march; etc. Some of the interactions will require protocol (and possibly database) extensions. The following list is preliminary and not necessarily complete. Further study and definition of requirements should be performed, and implications (bandwidth, etc.) investigated.

SCALE

An order of magnitude increase, approximately, in gaming area and in the number of simulated aircraft will be needed for fully exercising a single HIMAD fire unit simulator or trainer. This estimate is based on area of radar coverage and track load capacity. For a tactical organization of HIMAD embedded trainers (representing fire units at different sites, and their fire direction centers) the scale requirements will be even greater.

RADIATE PDU

The present Radiate Protocol Data Unit may be adequate for reporting information concerning each of the HAWK radars, except for its limit of 33 illuminated target IDs. The Radiate PDU is inadequate for PATRIOT's multi-function phased array radar, unless a PDU is issued every time a target is illuminated (one may be illuminated in search mode, the next in tracking mode, etc.) and this could produce an unacceptable PDU load.

ECM PDU

An ECM PDU is needed for reporting jamming actions and their characteristics.

IFF PDU

An IFF PDU (or possibly one each for interrogation and response) is needed for reporting IFF actions and their parameters.

AIRSPACE MANAGEMENT

HIMAD and friendly air units coordinate their activities by allocating the air spatially and temporally. If the HIMAD and TAC organizations participating in a SIMNET exercise extend to sufficiently high level, exchange of coordinating information can utilize tactical C3 channels. If no, airspace management information may have to be added to the "terrain" database with on-line changes via another new PDU.

Concerns of performing high fidelity ground vehicle engineering simulations with the proposed standard protocol and PDUs

Rosalind L. Spina
General Dynamics Land Systems Division
3800 Mound Road
Sterling Heights, MI 48310-3200

February 15, 1990

Position Paper:
Concerns on performing high fidelity ground vehicle engineering simulations with the proposed standard protocol and PDUs.

Rosalind L. Spina
Chief Engineer, Simulation Laboratory
General Dynamics Land Systems Division
38500 Mound Road
Sterling Heights, MI 48310-3200

Introduction:

There are several issues that are of concern in the use of the proposed standard protocol that could significantly limit the usefulness of any system for high fidelity engineering simulations. It is not the intent of this paper to propose solutions to these short comings, but only to voice them as areas in need of further thought and discussion.

Issues:

1. The SIMNET protocol does not include any protocol data units for digital burst communication. The future growth of inter-vehicle communication requires that map, map overlay, and echelon information be transferred in digital burst communication via radio. The protocol be flexible enough to handle this expanded requirement of the standard communications.

2. Pre-calculated trajectory data for projectiles is unacceptable for ground vehicle use. New projectiles require that the targeting vehicle not only give the initial launch location and termination location but also any other information required to calculate different trajectory profiles. This becomes very evident when the projectile is a "smart" round or does not fall into one of the normal trajectory paths.

3. The issue of determining hit and damage needs to be more flexible. When vehicles or munitions are classified or do not follow the norm for the system, hit and damage determination could be significantly affected. A vehicle with counter-measures which may have issued a decoy could be misread by the targeting vehicle. Additionally, new types of ammo could do more damage than normally expected. Hit and damage determination should be able to be tailored to the exercise, vehicle, and ammo type.

From: "Jerry Burchfiel" <burchfie@BBN.COM>
Subject: White Paper Submission
To: burchfie, GOLDIEZ%UCF1VM.BITNET@VTVM1.CC.VT.EDU, SIMNET-UCF@A.ISI.EDU
Cc: Apope, JAKane, Miller, SSmyth
Date: Thu, 15 Feb 90 18:48:50 EST

Response to "Position Paper on the Selection of a Global Coordinate System"

C.S. Smyth and J. Burchfiel, BBN-STC Advanced Simulation Division

February 15, 1990

A. General Comments

The use of a local topocentric coordinate system (the previous SIMNET standard) has been suggested (Ref. 1) as a possible basis for a global coordinate system in distributed simulation. Careful examination of that proposal confirms the notion that a topocentric system has several admirable qualities when used as an internal simulator coordinate system. Unfortunately, those local advantages would lead to unacceptable difficulties if such a coordinate system were employed as a global coordinate system.

Distributed simulations may involve any part of the region from subsea depths to geosynchronous satellite orbit. Individual simulators may represent areas ranging from small, possibly disjoint, surface patches to the entire near- earth volume. A desired near-term simulation capability is to conduct corps and echelon-above-corps exercises involving 10,000 to 30,000 vehicles operating across the entire European Theatre, thousands of miles in extent. Of course, the vast majority of these vehicles will be provided by semiautomated forces, not existing simulators.

A global coordinate system (e.g., that proposed in our previous white paper, Ref. 2) is required to communicate position and derivative information among the networked participants. The primary function of global coordinates is to specify absolute position unambiguously. They should do so with an economical use of space within packets and with a low cost to convert to and from internal simulator coordinates. They should also support low-cost conversion to common systems used for human interface, such as the MGRS, UTM, UPS, and geodetic systems other than WGS84. Accuracy of these conversions needs to be within a few meters over an entire theatre, so that simulated navigation equipment (e.g., GPS, the Global Positioning System), correlates within its positional accuracy to out-the-window views of landmarks identified on standard maps.

In contrast, a local coordinate system must meet the requirements of a particular simulator. The choice of a local system is a prerogative of the simulator manufacturer. Different simulators use different internal representations. In fact, many simulators use more than one internal coordinate system to better handle the geometrical relationships between the land surface, vehicles and their moving parts, and the replicated geometry of standardized structures.

Unfortunately, any local coordinate system that meets the special needs of a particular simulator (e.g., a topocentric system that makes the vertical and the positive Z-axis nearly coincident near the origin of the local area relevant that simulator) will impose three unnecessary

penalties on other networked simulators:

(1) the origin of each local topocentric system will have to be known by all participants (the origin is implicit in a global system),

(2) the local system will have to be converted to a new local system by all participants that do not share both a common proprietary internal representation and an identical gaming area, and

(3) an arbitrary simulator cannot log on to the simulation unless protocol extensions to describe the local system in use and a server to provide this information are added to the network.

The use of a single geocentric Cartesian coordinate system avoids these difficulties and allows a simulator to economically convert a global position specification to a wide range of specialized local coordinate systems.

B. Specific Comments on Accuracy of Coordinate Transformations

Seven coordinate transform equations are presented in Reference 1 which are claimed to "give precise results." Unfortunately, these equations assume a perfectly round Earth, characterized by the single constant "Rearth." In contrast, the WGS84 model of the Earth is an ellipsoid flattened at the poles by about 1/3%. This results in changes in surface curvature in the meridonal plane by about 1%, and differences in curvature between the meridonal and prime vertical planes of up to 2/3%. Use of a round Earth coordinate conversion model (constant Rearth) over a theatre-sized battlefield would result in unacceptable differences (multiple kilometers) between landmark positions on standard maps and readouts from simulated GPS or other navigational equipment.

More precise conversion algorithms, such as those given in Ref. 2, are required.

C. Conclusion

We recommend adoption of the Global Coordinate System described in Reference 2.

D. References

1. Rich Soeldner, Naval Training Systems Center, "Position Paper on the Selection of a Global Coordinate System", NSIA / DARPA / University of Central Florida Second Workshop on Standards for Interoperability of Defense Simulations, Orlando, Florida, January 1990

2. Jerry Burchfiel and Stephen Smyth, "Use of Global Coordinates in the SIMNET Protocol, White Paper ASD-90-10", NSIA / DARPA / University of Central Florida Second Workshop on Standards for Interoperability of Defense Simulations, Orlando, Florida, January 1990

-------- Jerry Burchfiel, (617) 873-3298

Jerry Burchfiel, (617) 873-3298

SIMNET Protocols as a Basis for Distributed Simulation Standards

Arthur R. Pope
BBN Systems and Technologies Corporation
February 14, 1990

Two recent papers by L. Michael Sabo ([1] and [2]) make several erroneous claims about the SIMNET protocols, and about distributed simulation in general. The following discussion identifies and examines each of those claims.

The papers [1] and [2] assert the following:

Erroneous Claim 1: The SIMNET protocols are proprietary to BBN.

The SIMNET protocols were developed for the U.S. Government, and they are entirely Government–owned. BBN does not, in any sense, have a proprietary position in the SIMNET protocols.

The U.S. Government has chosen to offer the SIMNET protocols as a basis for a distributed simulation standard. The protocols are fully described in the report The SIMNET Network and Protocols [3], which has been widely distributed. That report provides all information needed to implement the SIMNET protocols, including a complete definition of every bit communicated among simulators.

Erroneous Claim 2: "A specialized hardware protocol converter" is required to implement SIMNET protocols.

SIMNET protocols can be implemented readily using commercially available hardware. Indeed, SIMNET simulators have been constructed from various platforms, including personal computers and UNIX workstations. Some of these simulators have used common Ethernet interface processors to carry out low–level protocol processing. The implementor has many options without resorting to custom hardware.

Some existing simulators have been connected to SIMNET networks via protocol converters or external processors. In these cases, additional hardware was used for convenience. This approach ensured that (a) there would be minimal modification to the existing simulators, and (b) the processing of SIMNET protocols would be contained in a well–partitioned, and therefore reusable, module.

Erroneous Claim 3: The SIMNET protocols are not based on an open communication architecture.

By "open communication architecture", Sabo presumably means an architecture that permits the interconnection of simulators produced by various manufacturers. The SIMNET protocols are based on such an open architecture. Since the SIMNET protocols are not private, and since no special hardware is required to implement those protocols, any manufacturer can build a simulator for inclusion in a SIMNET network. Moreover, because the SIMNET protocols are defined within the framework established by the ISO Basic Reference Model for Open Systems Interconnection, those protocols can use standard communication services, protocols, and products whenever they are available and appropriate.

Erroneous Claim 4: The SIMNET association protocol is not an application layer protocol.

ISO International Standard 7498 [4] defines the Basic Reference Model for Open Systems Interconnection, including the purpose of each of the model's seven layers. It says (§7.1.4) that "the Application Layer contains all functions which imply communication between open systems and are not already performed by the lower layers." In accordance with this definition, the association protocol resides in the application layer, where it provides functions not already performed by the lower layers.

What are those functions that the association protocol provides? The functions include some things typically found in session and transport layers, such as a mechanism for blocking multiple messages into single transmissions and a mechanism for treating certain messages as either requests or their associated responses. However, the association service must differ from standard session and transport services in an essential way: it must provide efficient delivery of data to multiple destinations.

Distributed simulation requires a communication service that can efficiently convey a single message to multiple recipients. This form of communication, called *multicasting*, is used to disseminate information about vehicles and events in the simulated world. There are some inefficient ways of multicasting; one can, for example, transmit a separate copy of a message to each of that message's recipients. However, it is important that multicasting be performed efficiently since nearly all of the messages communicated in a distributed simulation must be multicast quickly to all participating simulators. Fortunately, some popular (and standard) local area network technologies provide efficient multicasting services.

To date, ISO efforts to define standard transport and session services have focused on communication between a single pair of entities. ISO has not yet defined standard transport and session services that provide multicasting. And even though a network service may

provide multicasting, the overlying transport and session services provide no access to that capability, thus rendering it unavailable. If one uses the existing, standard transport and session services, one can only "multicast" a message by transmitting a separate copy to each recipient. We have created an association protocol that can provide the needed services in an efficient manner by taking advantage of a network service's inherent multicasting capability.

Why is the association protocol defined as an application layer protocol, rather than as a session or transport protocol? For the sake of efficiency, the association protocol has been designed to include precisely those features necessary for the support of distributed simulation. It contains some session-like features, and some transport-like features. By combining these features into a single package, the association protocol is able to implement them in the simplest, most efficient manner. It is neither a complete session protocol, nor a complete transport protocol, so it cannot correctly be called either. But because it provides services not already available from lower layers, we have found it most appropriate to view the association protocol as an application layer protocol. This view is compatible with the Basic Reference Model.

Erroneous Claim 5: The SIMNET protocols preclude use of standard transport and session services.

In the previous section, we explained why existing, standard transport and session services were not appropriate for distributed simulation. The critical ingredient missing from those services is support for multicasting.

However, the SIMNET protocols do not preclude the use of any transport and session services, as long as those services include essential features such as multicasting. Work on standard transport and session services continues, and some day those services will include the elements needed for multicasting, in accordance with Addendum 2 (Multipeer Data Transmission) to the Basic Reference Model [5]. When that happens, there will be no architectural obstacles preventing adoption of the new services. The SIMNET protocols, residing in the application layer, will be able to operate using the new, standard transport and session services.

Erroneous Claim 6: The SIMNET protocols are limited to using a particular network technology (Ethernet).

The SIMNET protocols are not dependent on the use of any particular network technology.

The protocols are defined in terms of a network layer interface, which is completely and concisely defined in the report The SIMNET Network and Protocols [3]. Today, there are available various network technologies that can provide the services required at that

network layer interface. Ethernet and FDDI are just two of the technologies meeting the stated requirements.

Erroneous Claim 7: The SIMNET protocols violate the OSI Basic Reference Model.

The SIMNET protocols are defined in terms of the OSI Basic Reference Model. The ISO International Standard that defines that model [4] states that, "the Reference Model serves as a framework for the definition of services and protocols which fit within the boundaries established by the Reference Model." This is precisely the manner in which the model has been used in defining the SIMNET protocols. There are no areas in which the SIMNET protocols violate this framework.

Please see Attachment A to this paper, a memorandum from Alex McKenzie to Duncan Miller, for further discussion of the Basic Reference Model.

Erroneous Claim 8: Distributed simulation protocols should be defined and encoded using ISO Abstract Syntax Notation One (ASN.1).

There are two parts to the ASN.1 definition: a notation, or *abstract syntax*, for defining data types, and a *transfer syntax* for encoding instances of those data types for transmission. ISO International Standards 8824 [6] and 8825 [7] describe those two components of ASN.1. The notation and transfer syntax have been separated so that, conceivably, the notation may be mapped to alternate transfer syntaxes. To date, only one transfer syntax has been declared a standard by ISO; that syntax is called the *Basic Encoding Rules*.

ASN.1 is a good choice for many application layer protocols. It provides a powerful and flexible method of describing and encoding elaborate data types. However, ASN.1 transfer syntax is not well suited for applications where efficiency is of great importance, such as distributed simulation. There are two reasons why SIMNET PDUs should not be encoded according to the Basic Encoding Rules. First, under that encoding scheme those PDUs would be considerably larger. Each PDU field and each group of fields would carry additional bytes describing their length and type. As a consequence, for example, a Vehicle Appearance PDU would grow from 132 bytes to about 215 bytes. Second, a significant amount of processing is required to encode and decode ASN.1 transfer syntax. ASN.1 fields do not reside at fixed positions within their PDUs. An ASN.1 field containing an integer, for example, occupies one or more bytes according to the size of that integer. As a result, the size of a field may vary among instances of the PDU, as will the positions of subsequent PDU fields. Software must encode and decode ASN.1 transfer syntax by scanning each PDU from beginning to end while copying data to or from fixed format data structures. In contrast, SIMNET PDUs are represented according to rules that make these encoding and decoding processes extremely simple and efficient.

It is especially important for real–time, distributed simulation that communication among simulators be efficient. SIMNET nodes must be able to receive and process hundreds (perhaps eventually thousands) of PDUs per second. We believe that the use of ASN.1 transfer syntax for distributed simulation protocols would significantly increase the cost and complexity of simulators and their communication networks without providing offsetting advantages.

Perhaps the ASN.1 notation can be mapped to an alternate transfer syntax that emphasizes conciseness of representation and efficiency of processing. Unfortunately, the present ASN.1 notation is closely tied to the particulars of the Basic Encoding Rules, embedding details of how data types are encoded according to those rules. We hope that some day there will be a standard notation with an accompanying set of encoding rules optimized for efficient, real–time communication (perhaps an "ASN.2"). When such an "ASN.2" becomes available, we believe it should be used for defining distributed simulation protocols, provided it is appropriate.

Conclusion

The SIMNET protocols are defined in terms of the framework established by the ISO Basic Reference Model for Open Systems Interconnection. Thus the SIMNET protocols can make use of ISO–standard communication services and protocols that are both appropriate and available.

There do not presently exist standard protocols at intermediate layers of the Basic Reference Model possessing the features needed for distributed simulation. This does not prevent the use of SIMNET protocols since they require only a network service, and that service can be provided by various local area network technologies. Nevertheless, the SIMNET protocols will be able to make use of appropriate standard protocols at the intermediate layers when those protocols become available.

The SIMNET protocols can be readily implemented in a variety of ways using common, commercially available hardware. All information required to do so is owned by, and has been published by, the U.S. Government.

References

[1] L. Michael Sabo; *Distributed Simulation Architecture*; Position paper for the Second Workshop on Standards for Interoperability of Defense Simulators, 26 January 1990.

[2] L. Michael Sabo; *Interactive Simulation Protocol*; Position paper for the Second Workshop on Standards for Interoperability of Defense Simulators, 31 January 1990.

[3] Arthur Pope; *The SIMNET Network and Protocols*; BBN Report Number 7102; BBN Systems and Technologies Corp; Cambridge, Mass.; July 1989.

[4] International Standards Organization; *Information processing systems — Open Systems Interconnection — Basic Reference Model*; ISO 7498–1984.

[5] International Standards Organization; *Information processing systems — OSI Reference Model — Part 1: Basic Reference Model — Proposed Draft Addendum 2 to ISO 7498–1 on Multipeer Data Transmission*; ISO 7498–1/PDAD 2.

[6] International Standards Organization; *Information processing systems — Open Systems Interconnection — Specification of Abstract Syntax Notation One*; ISO 8824.

[7] International Standards Organization; *Information processing systems — Open Systems Interconnection — Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1)*; ISO 8825.

MEMORANDUM

TO: Duncan Miller

FROM: Alex McKenzie

SUBJECT: The ISO Reference Model for Open Systems Integration

DATE: February 14, 1990

You have given me a copy of a paper titled "Distributed Simulators Architecture (DSA)" dated January 26, 1990, by L. Michael Sabo for review. You have asked me to comment on Mr. Sabo's proposal to develop a Distributed Simulators Architecture which "will follow the OSI model."

1. My Qualifications

   As you know, I have been involved in the standardization of communication protocols for open interconnection of heterogeneous systems for 20 years. I was a member of the "Network Working Group" which defined the ARPA Network Control Protocol (predecessor to TCP), Telnet Protocol (virtual terminal), and File Transfer Protocol; I was the author of the document which officially defined the first of these, and a co-author of the official documentation of the others. I have been a member of the International Federation for Information Processing (IFIP) working group on Architecture and Protocols for Computer Networks since 1973; I served as Chairman from 1979 to 1982 and as Secretary from 1982 to the present. I was one of 4 co-authors of a protocol which IFIP submitted to CCITT and ISO for consideration as an international standard, and I was active in the CCITT working group that drafted the X.25 protocol. As an IFIP representative I was present at the very first meeting of the ISO committee which was charged with developing standards for OSI.

   From 1981 through 1986 I was in charge of a series of BBN contracts with NBS (now NIST) to assist in the development of US government positions on the specification, design, and evaluation of protocol standards being established by national (ANSI) and international (ISO) standards-making organizations. I directly participated in the development of standards for the Presentation layer (layer 6) of the ISO Reference Model for OSI and, in fact, served as the chairman of the ISO group responsible for the Presentation service and protocol (including ASN.1) from December, 1983 through November, 1984. Additionally, I supervised other BBN staff members working on ANSI and ISO standards for Network Layer, Transport Layer, Session Layer, File Transfer, Virtual Terminal, and Formal Description Techniques.

2. ISO's Basic Reference Model and OSI Standards

   It is vital to understand the purpose of ISO in developing International Standard 7498 (the Basic Reference Model). The purpose is described in Clause 1, "Scope

and Field of Application" which is so short it is worth reproducing here in its entirety:

"This International Standard describes the Reference Model of Open Systems Interconnection. It establishes a framework for coordinating the development of existing and future standards for the interconnection of systems and is provided for reference by those standards.

This International Standard does not specify services and protocols for Open Systems Interconnection. It is neither an implementation specification for systems, nor a basis for appraising the conformance of implementations."

In the first paragraph above, it is stated that the Basic Reference Model exists as a framework for development of standards. ISO recognizes *itself* as the cognizant standards-making body in this field. In other words, the Basic Reference Model is intended for ISO's *internal* use. One might ask, if this is the purpose of the Reference Model, why does it need the status of an International Standard? The answer to that question is contained at the end of the first paragraph; it "is provided for reference by those standards," since those standards will best be understood within the context of a self-consistant framework or model.

The second paragraph says what the Basic Reference Model is **not**. It does not specify services, nor does it provide "a basis for appraising the conformance for implementations." There is no conformance clause in ISO 7498, as there is, for example, in ISO 8823 (Connection Oriented Presentation Protocol Specification). Thus, in a formal sense at least, it is meaningless to talk about whether some given architecture does or does not "conform to the ISO Basic Reference Model" for Open System Interconnection.

To put it a different way, one could invent an infinite number of "protocol stacks," each one of which meets the criteria given in ISO 7498; one could then argue that each stack conforms to the Basic Reference Model. But doing this would in no way meet the goal of ISO to provide " a small number of practical subsets to facilitate implementation and compatibility" [clause 0.1 of ISO 7498]. The interconnection of systems will only be facilitated if they implement the *same protocols,* not different protocols based on the same model. The GOSIP "announcement section" [Federal Register, Vol. 54, No. 133, page 29598] expresses this same concept as follows:

"GOSIP defines a common set of data communication protocols which enable systems developed by different vendors to interoperate and enable the users of different applications on these systems to exchange information. These Open Systems Interconnection (OSI) protocols were developed by international standards organizations, primarily the International Organization for Standardization (ISO) and the Consultative Committee on International Telephone and Telegraph (CCITT).... The primary objectives of this standard are:

- To achieve interconnection and interoperability of computers and systems that are acquired from different manufacturers in an open systems environment;

- To reduce the costs of computer network systems by increasing alternative sources of supply;"

3. The "Distributed Simulators Architecture (DSA)" proposal

In section 4 of the DSA proposal, it is recommended that a DSA architecture which "will follow the OSI model" be developed. It is implied, although not actually stated, that developing a DSA which "follow [s] the OSI model" is better for the government than using an architecture which does not "follow the OSI model." However, the proposal also recommends that working groups be set up to *define* protocols for each of the Physical, Datalink, Network, Transport, Session, and Presentation layers (see Figure 3 of the proposal). This approach will actually derive no benefit whatsoever from the hundreds of person-years of work which have gone into the definition of ISO Standard protocols at each of these layers. This approach, while paying lip service to the standards process by "follow [ing] the OSI model" will not benefit from commercial implementations of ISO protocols. Instead, it will increase both the cost (through support of the DSA committees and working groups) and the delay in obtaining simulator interoperability.

In my opinion, the needs of the government will best be served by defining an Application Layer protocol for distributed simulations which assumes the existence of real-time multicast service from the ISO Presentation layer, and then waiting until that service is actually available, providing an ad-hoc solution in the meantime. This allows the design of real-time multicast protocols in each layer to be carried out by the international experts working on those layers, and minimizes the government's investment in specifying and building machinery which will have to be discarded when the international standards come into being.

POSITION PAPER

Network Topologies for a Unified Simulation Internet

Reference: 131.rsc.208.19

Submitted by:
Michael Robkin
Randy Saunders
Hughes Simulation Systems, Inc.
1200 E. San Bernardino Road
West Covina, CA 91790

ABSTRACT

This paper describes the applicability of the SIMNET network model to various simulation internet topologies.

INTRODUCTION

The development of a standard for networking simulators must address a number of diverse topics, central topics among which are: (a) the architectural model used to define the network, (b) the communication model used to define how network entities interact, (c) the form and function of network interactions, and (d) the representation of network information in a standard format. This paper presents a discussion of a architectural and communication models that support a wide spectrum of future networking applications.

The SIMNET architecture developed by BBN Systems and Technologies has been offered as a potential baseline for a standard for networking simulators. Two open conferences on the subject have surfaced a number of different issues bearing upon a potential standard. With due appreciation to the above sources, this paper describes a different architecture in order to generalize the simulation internet.

SIMNET ARCHITECTURE

The SIMNET architecture is curious indeed. It is a single level of machines that are full peers, with each machine expected to both provide everything that any peer needs, and process everything produced by all peers. The basic information unit contains all the information known about the sender.

To obtain efficiency adequate to support a few hundred slow maneuvering vehicles, the SIMNET system utilizes direct multicasting onto a dedicated Ethernet channel. The result is well-suited to the SIMNET requirement to support a number of essentially identical units at minimum cost. However to meet future needs, which cannot be accurately predicted, a more open architecture that can support many types of computers is required.

SIMULATION INTERNET ARCHITECTURE

In this paper we propose a Simulation Internet Architecture (SIA) which provides an alternative baseline to that posed by SIMNET. The first characteristic of our proposed SIA is the separation of the network topology from the standard. Any standard should define the interface between a single simulator at its network interface port, not the entire network as an entity. We recommend that the connectionless nature of SIMNET be maintained without the presumption of multicast to all entities. Rather, we propose that the application layer protocol include query Protocol Data Units (PDUs) in addition to the SIMNET PDUs, which only state attributes of the sender. As a direct result, it will become possible to add distinguished entities who provide information to all simulators, such as weather, which individual simulators do not have the problem-wide scope to control. Such entities, although useful in connecting different types of simulators, are conceptually prohibited by the SIMNET philosophy that the network consist only of equal peers. In practical operational situations, this philosophy is simply not affordable.

It is important to note that standardization on our SIA does not preclude the existence of SIMNET-like implementations. In the special case in which all simulators are identical and capable of obtaining everything they need from what each other sends, outside information sources are not needed. Also, the inclusion of queries in the standard does not mean that the queries cannot be answered by the simulator about which they need information. The result is that SIMNET is an interesting special case of our proposed architecture, and in certain small problems it could be chosen as the most effective solution via engineering design tradeoffs. We view this as a favorable result, in that it allows acceptance of the fundamental soundness of the SIMNET approach while allowing its functionality to be extended.

One negative aspect of the SIMNET approach is its introduction of association and data collection protocols that have little to do with the application layer interface of a simulator to its network interface port. We propose that all these non-simulation protocols be eliminated from any standard. The existence and interface to these types of operations is not a suitable topic for an interoperability standard. Operations addressed by these protocols ought rather to be handled by network internal mechanisms standardized separately as part of GOSIP or whatever protocol stack on top of which the SIA application layer protocols operate.

COMMUNICATIONS MODEL

Three basic operations establish the application layer interface between a simulator at its network interface port:
(a) the port may provide some data present on the network;
(b) the simulator may provide state information about itself for presentation on the network; and (c) the simulator may provide a query describing data that it desires. All these operations are

asynchronous; neither their delivery nor order preservation is assured by the network. The network merely accepts data, accurately stamps it with a precise time of presentation and them makes its best attempt to deliver it.

It is likely that the box that implements the network interface port might perform certain intelligent operations. For example, it might use the last reported simulator position to select a subset of the information on the network for presentation to the simulator. It might use traffic analysis information to determine which of a multiple of Ethernet or FDDI lines to transmit its data on. It might maintain internal copies of some network data so that it could respond to future queries without producing any network traffic. It might use internal data to pad a brief network PDU out so that it contains all the information required by a SIMNET module. These many types of intelligent network operations must be considered part of the network design, and they are wholly inappropriate for inclusion in an interoperability standard because they require design tradeoffs that cannot be made once and for all without excluding a significant number of applications.

Certain situations will benefit from the use of a connection-based protocol extension. Among the applications for which SIMNET currently uses such a protocol for is the remote initialization of a simulator from the System Control Console (SCC). We believe that this is not a good topic for standardization, although nothing we propose would prevent this from being done. We view this decision as a design one that each developer should be free to make in that there is no interoperability goal in this area.

We do not believe it is either desirable or likely that a user would want to use a SIMNET SCC to initialize an existing F-18 weapons simulator that already has an instructor/control station. Although it makes fine sense for a SIMNET SCC to use the network to initialize positions and similar functions, the flight simulator control station directly wired into the main computer would be needlessly constrained if forced to use the network. Most significantly, by using a strict application layer model for the simulation protocol, the addition of a connection-based initialization protocol costs nothing for those who do not use it. The flight simulator would require no modification in response to initialization protocol changes made in the SIMNET SCC. This further argues against standardization of simulator specific operations such as initialization. When these protocols are added, they can use lower-layer connection protocols to connect to individual simulators and negotiate as needed. Once a one-to-one connection is established, the dialog should be strictly up to the two simulators, in our opinion.

SECURITY

Security is a significant concern for the integration of weapon system simulators into the network. Use of simulators to evaluate weapons and/or establish tactics and doctrine elevates security concerns to a critical level. Security evaluation of simulator software would impose an unacceptable cost and schedule impact just to support networking. External devices will be required to implement security policy. To remain compatible with these devices, the simulation protocol must be an application layer protocol. Security protocols defined for such activities as key management already have application layer interfaces. Disrupting their assumptions about the protocol stack would invalidate their assurance proofs and inflict significant evaluation costs

RECEIVED NETWORK DATA PDUs

Three types of information are presented to the simulator: time stamp of when the data was produced; time stamp of when the data was given the simulator; and PDU contents as defined in the next two sections. With respect to time stamp concepts, we support the 32-bit integer time stamp format proposed and defined by Dr. A. Katz in position paper 008-01-90.

OWN STATE INFORMATION PDUs

Own state information PDUs follow the basic concepts of application layer PDUs presented in SIMNET and various position papers. A complete description of the form and representation that we recommend is presented in our other position papers, 131.rsc.209 and 437.mrc.100.. Most naming changes are presented only to avoid the term vehicle, in that this term applies to no more than half the entities in a simulation and its application to such things as artillery rounds in flight and drifting clouds of smoke adds more confusion than is acceptable in a standard for the future. Words that appear inside angle brackets are specific terminal fields in the PDUs, as defined in 131.rsc.209 (e.g. <sample terminal>).

Identification PDU. - The Identification PDU provides information about the entity that never changes, such as its <ID number> and <entity type>. The transmission of this PDU with a previously unseen <ID number> constitutes the creation of a new entity.

Configuration PDU. - The Configuration PDU provides information about the entity that represents changes in its appearance for which an understanding of its <entity type> is required. For an M2 vehicle, this would include the position of the TOW launcher. For an M1 tank, this would include the position, rotation rate and elevation rate of the gun. For an F-18 aircraft it would include the stores visible on each wing pylon. Firing a weapon having visual effects results in a new configuration PDU.

Position PDU. - The Position PDU provides information about the entity that is independent of its <entity type>. It includes its <position>, <velocity>, <orientation>, <rotation rate>, and <articulations>.

Explosion PDU. - The Explosion PDU indicates the release of a certain amount of energy at a given location. This PDU applies primarily to munitions and missiles.

Damage PDU. - The Damage PDU provides the same information as the Configuration PDU which resulted from the explosion of an entity whose <ID number> is contained within the PDU. The differentiation between this and the Configuration PDU is to support the presentation of damage to an entity in a simulator that does not fully support its <entity type> with imagery/symbology for all configurations. It also may be used by a weapon simulator to verify that an explosion PDU was understood, as some weapon simulators might have the capability to cause their network interface to repeat explosion PDUs to assure delivery.

Property PDU. - The Property PDU is used to convey a specific named property of the entity. The property is identified by the <property string> upon which the meaning of the values transmitted depends.

Error PDU. - The Error PDU is used in response to a query type of PDU to which the identified entity cannot otherwise respond. It indicates that the entity does not know the desired information. The transmission of an error PDU might signal a distinguished entity to provide default information. It is thus possible that a query type of PDU could result in both an error response and a data response, in which case the data response should be used.

QUERY PDUs

Query PDUs follow the same basic concepts as the other application layer PDUs presented above. A complete description of the form and representation that we recommend please refer to our other position papers, 131.rsc.209 and 437.mrc.100. Most simulators will want to utilize an algorithm for determining who ought to respond to the "all" and "general" PDUs described below. Although we feel no such algorithm ought to be required, the waste of network bandwidth that would result if every simulator responded to a general query will require the use of some algorithm. Dissimilar simulators can be interconnected more readily if an additional arbitration algorithm is not placed on both. A few duplicate messages is a reasonable price to pay in exchange for greater flexibility. The use of "all" PDUs may be discouraged by some network administrators. The recommended way to come up to speed on an exercise in progress is to determine which entities are appropriate and then send specific query type PDUs to them.

Query Identification PDU. - The Query Identification PDU asks the specific entity whose <ID number> is provided to transmit an Identification PDU.

Query Configuration PDU. - The Query Configuration PDU asks the specific entity whose <ID number> is provided to transmit an Configuration PDU.

Query Position PDU. - The Query Position PDU asks the specific entity whose <ID number> is provided to transmit an Position PDU.

Query All Identification PDU. - The Query All Identification PDU asks all entities to transmit an Identification PDU.

Query All Configuration PDU. - The Query All Configuration PDU asks all entities to transmit a Configuration PDU.

Query All Position PDU. - The Query All Position PDU asks all entities to transmit a Position PDU.

Query Property PDU. - The Query Property PDU asks the specific entity whose <ID number> is provided to transmit an Property PDU with the specific <property string> provided.

Query General Property PDU. - The Query General Property PDU asks that any entity who knows the value of the specific <property string> provided to transmit an appropriate Property PDU.

Query General Location Property PDU. - The Query General Location Property PDU asks that any entity who knows the value of the specific <property string> provided at the specific location to transmit an appropriate Property PDU. This PDU allows a simulator to query information about a specific location to update or extend its terrain database.

CONCLUSION

While the SIMNET protocols have laid the foundation for a simulation internet, better architectures and communication models can improve its achievability. Extensions to support queries are desirable to more easily support diverse networks. The fundamental concepts of dead-reckoning and providing a minimum level of interaction with unfamiliar simulators are supported and continue to provide the same functionality they do in SIMNET.

POSITION PAPER

Notations and Units for a Unified Simulation Internet

Reference: 437.mrc.100.26

Submitted by:
Michael Robkin
Randy Saunders
Hughes Simulation Systems, Inc.
1200 E. San Bernardino Road
West Covina, CA 91790

ABSTRACT

This paper defined the specific representation and units for values transmitted within the simulation internet. The objective of this system of units is to provide: (a) sufficient accuracy that user-perceptible jumps do not occur, and (b) representations using a minimum amount of network traffic, both without jeopardizing real-time performance.

INTRODUCTION

The development of a standard for networking simulators must address a number of diverse topics, central among which are: (a) the architectural model used to define the network, (b) the communication model used to define how network entities interact, (c) the form and function of network interactions, and (d) the representation of network information in a standard format. This paper presents the notation and units of values used in the Protocol Data Units (PDUs) presented in our position paper reference 131.rsc.209.

The SIMNET PDUs and their notations developed by BBN Systems and Technologies have been offered as a potential baseline for such a standard. Two open conferences on the subject have surfaced a number of different issues bearing on a potential standard. With due appreciation to the above sources, this paper describes a different set of notations and units in order to generalize the simulation internet.

UNIT SELECTION STRATEGIES

The selection of units and notations for a MIL-STD must take into account the long-range effects of the decisions. The objective ought not to be simply to select the best of today's ideas but also to assure that future better solutions are not excluded. Given the pace of change in the computer world, this is a tall order, but we believe the recommendations of this paper address likely long-term issues.

First and foremost, we recommend AGAINST the use of any floating-point numbers. Although the IEEE floating-point standard is very popular, it is far from universal. Many companies make machines that perform best with another floating-point standard. The most notable of these other floating-point formats is the Digital Equipment Company format used in DEC/VAX computers, as well as coprocessors and accelerators for the VAX made by other companies such as Applied Dynamics and Floating Point Systems. To exclude the entire VAX family of computers, or to reduce their performance by requiring time-wasting conversions, seems counterproductive. Even if everyone were to standardize on format, the cost of floating-point instructions is still many times that of integer equivalents. This means that more powerful computers will be required to provide network compatibility and reducing the breadth of implementation of the standard.

All numbers required for the PDUs we recommend can be expressed in simple integers. Computers with excess floating-point instructions can perform single multiplies to convert the units we propose into whatever units they choose, while computers without the excess capacity can perform quicker integer operations. The benefits are available to all systems, while the time impact of floating multiplies for conversion is carried only by those machines possessing the excess capacity.

Using only integer operations makes it more feasible to retrofit network capabilities into existing simulators. The only advantage of floating-point numbers is the speed with which design can be accomplished, because limits and accuracies need not be addressed before code can be written. Thanks to SIMNET, the architectural issues have been investigated and the principles proven leaving the simple task of investigating and selecting units.

Second, we support the SIMNET decision to use only SI units in that the conversion to SI (metric) units brings the standard into alignment with the broadest scientific community. The notational choices made result in non-integral scales for some units, but these notations fall out of measuring angles by the length of their surface arc and the radius of the earth is not even in metric or English units.

Third, we recommend measuring all angles with Binary Angle Measurements (BAMs), the unit used to measure turret rotation in SIMNET. We use two sizes of BAMs, depending on the accuracy required. 32-bit BAMs (<BAM32>) divide the 2*pi circle into 2**32 units. 16-bit BAMs (<BAM16>) divide the 2*pi circle into 2**16 units.

GENERAL POSITION MEASUREMENT

We recommend the Binary Angle Measurement of Latitude and Longitude (BAMLL) described by Glasgow in "World Coordinate System," but with a few modifications. A brief description of BAMLL follows.

Any position on or near the surface of the earth would be described by latitude, longitude, and distance from the center of the earth: LON, LAT, and R.

Longitude would be measured by dividing 360 degrees into a 32-bit integer number. This gives an accuracy of 1 bit = 9.3306mm on the surface of the earth at the equator. The lines of longitude move closer together nearer the poles so that, at 60 degree North latitude, the longitude resolution is 1 bit = 4.7mm. Greenwich would be at 0, and East the positive direction.

For simplicity latitude would be scaled the same so that 1 bit also = 9.3306mm on the surface of the earth. 0 degree = 0 bits at the South pole, and 180 degrees = 2**31 bits at the North Pole.

Using the same accuracy for R, 1 bit = 9.33mm, the largest 32-bit integer value that R could represent would be 40 million meters, over six times the radius of the earth. This choice is arbitrary, R could as well be defined in millimeters from the DMA ellipsoid, the geoid, or even some sphere. Our recommendation is based on the fact that most common simulators use cubic units that are similar in X, Y, and Z. The value of similar accuracies was considered of relative importance. For lack of any good term for this unit of distance measure, we have chosen the BAM32+ symbol to indicate the length that a one BAM32 arc subtends at the equator. An unsigned 16-bit number expresses a distance in BAM32+ units covering over 600 meters. This allows worldwide measurements to be made at course, 600 meter, resolution using a 16-bit number. Similarly, short distances of under 600 meters can be measured using a 16-bit number.

ADVANTAGES

First, BAMLL converts directly into the common latitude and longitude coordinate system. Although this may not have any real-time computational advantages, it does make conversion to and from existing mapping data a staight forward process.

Second, any global position can be converted to a local topocentric position and vice-versa without any complex matrix conversion calculations. Since many existing simulators use a topocentric Cartesian coordinate system, they could interface with a BAMLL global coordinate system without major modifications.

Over 50km distances typically used in training scenarios, the error in position between the global coordinate system and that of a topocentric Cartesian coordinate system is around 1%. This is adequate for many simulations and no corrections will be necessary. Over small distances (1km) the error in distance is 0.001%, or about 1 BAM32+. Therefore, if corrections in position are required, they will not have to be computed continuously. Updating one's position every kilometer traveled will produce jumps of no more than the 1 BAM32+ being reported, and so they will not be visible.

Exact formulas for determining position, [illegible], and range for all cases have been published in a variety of sources, including "Position Paper on the Selection of a Global Coordinate System" by Rich Soeldner.

A simple formula for converting BAMLL position to a topocentric Cartesian position is:

Y - Yref[mm] = 9.33[mm per BAM] * (LAT - LATref)

X - Xref[mm] = 9.33[mm per BAM] * (LON - LONref) * sin(LAT)

assuming LAT and LON measured as described above. It should be noted that sin(LAT) in the above formula does not require calculation of a trig function very often. The sin() function changes most rapidly at the poles, not the most common spot, and at 90 degrees north a 0.01% change in value requires a change of over 20km in position. This would be accurate enough for many simulations, and would allow future simulators to use the global position standard internally, where a topocentric Cartesian coordinate system is often used now.

Third, by representing position as integers, the time for mathematical calculations is decreased. Although the actual time used by a processor to perform calculations is dependent upon the implementation, algorithm, compiler, and host processor, it has been our experience with a variety of processors that the same calculations can be completed in much less time using integer rather than floating-point calculations.

Other time savings are possible when angles are measured in integer values. In some of our microprocessor simulators using BAMs we have used look-up tables in place of Sin and Cos calculations. A Sin and Cos lookup table of 32-bit values accurate to 15 seconds of arc requires 86 Kbytes of memory, not a great deal at current DRAM prices. Lookup tables are much faster than any transcendental Sin functions, but actual time savings depend upon the implementation and microprocessor.

Fourth, many range calculations can be performed in two dimensions. The difference in altitude for most naval, ground, or air simulations is negligible compared to range in the X and Y direction. If, for example, a simulator was sorting out all other simulators' positions at ranges more than 5 km away, a two-pass range calculation could be implemented. In the first pass, only a rough range calculation is necessary, eliminating the majority of inputs before the second pass calculates an exact range.

The first pass, using the formula

abs(X - Xo) >= Range   or   abs(Y - Yo) >= Range

to compute a rough range value, requires only two calculations, two sign manipulations,two comparisons, and one Boolean operation.

The first pass formula, as suggested by Burchfiel and Smyth in "Use of Global Coordinate in the SIMNET Protocol" to do range calculation in 3 dimensions using his proposed Geocentric Cartesian system, is:

$X < Xo - R$ or $X > Xo + R$ or $Y < Yo - R$ or $Y > Yo + R$ or $Z < Zo - R$ or $Z > Zo + R$

It would require six calculations, six comparisons, and five Boolean operations. Clearly our method using BAMLL positions is more efficient.

Fifth, the data size of BAMLL (3 x 32 bits = 96 bits) is one-half of the 192 bits required by the Geocentric Cartesian coordinate system proposed by Burchfiel. That implementation would require three 64-bit floating-point numbers. Using BAMLL would immediately save 96 bits of position data.

GENERIC TYPE IDENTIFICATION

Evaluation of the SIMNET notations has raised several issues related to the representation of different types of objects associated with various organizational roles. The primary concern is that a medium-fidelity tank simulator need not know the exact configuration of an aircraft to represent it as fully as it can. Knowing that it is a fighter and either friend or foe is probably sufficient to provide the tank commander with an adequate image. In contrast, a sophisticated EW simulator may need to know what version of anti-air missile the fighter is carrying to present correct displays so that its operator chooses the correct countermeasure. The typing system must be accurate enough to support the EW simulator without requiring the tank simulator be updated whenever a new missile is introduced.

We believe that no encoded numerical notation can do the job adequately, and we propose the introduction of a hierarchically structured string designation to support continuous expansion of detail in these situations. The string would be delimited into blocks by "." characters. Each simulator would be free to parse out as many fields as it could understand. Due to the length of these strings, their static nature would be used to transmit them only once when the entity comes on-line.

SPECIFIC FIELD TYPES

The fields in the PDUs presented in our position paper reference 131.rsc.209 were defined in a notation- and units-independent way. This paper recommends units and notation for each field type.

<[illegible]> - Identifies the simulation network entity.

```
type Simulation_Network_ID is
  record
    site : <number>;
    host : <number>;
    entity : <number>;
  end record;
```

<entity type> - Identifies type of network processing the entity receives. This has little to do with what type of machine it is, as such diverse weapons as tanks and patrol boats have the same network needs.

```
type Entity_Kind is (Irrelevant_Class,  -- None of the others
       Static_Class,                    -- Does not move or shoot
       Platform_Class,                  -- Does not move
       Mobile_Platform_Class,           -- Moves, not articulated
       Articulated_Platform_Class,      -- Complex motion
       Invisible_Observer_Class,        -- Unseen point of view
       Airborne_Class,                  -- Does not shoot
       Airborne_Platform_Class,         -- Normal aircraft
       Projectile_Class,                -- Ballistic round(s)
       Missile_Class,                   -- Non-ballistic weapon
       Cloud_Class,                     -- Smoke or NBC clouds
       Flare_Class,                     -- Point light source
       Emitter_Class,                   -- Point EM source
       Sensor_Class);                   -- Sensor capability
```

<time stamp> - Identifies time after the hour as defined by Dr. A. Katz in his position paper.

```
type Time_Stamp is integer range 0..4294967295;
```

<weapon system> - Identifies make and model of weapon system that the entity simulates.

```
type Weapon_System_Descriptor is string(0..512);
```

The string would contain at least six hierarchical demarcations, Defined as follows:

1. Country of design
2. Country of deployment
3. Armed Forces branch of deployment
4. Operational function code (assigned by branch)
5. Model
6. Version
7. Block or subversion
8. Configuration
9. Optional subsystems, as many as exist
10. Serial number

[illegible] examples [illegible]
The USAF night-capable F-16 could be
"USA.USA.USAF.F.16.C.Block 40.Night.LANTIRN.HARM.Tail 90-550".
The Navy Aegis cruiser Antietam could be
"USA.USA.USN.CG.Aegis.54...TOMAHAWK.SQS-53B.SQR-19.
SQQ-28.Antietam".
The M1 Abrams Main Battle Tank could be
"USA.USA.USA.M.1.A1..Platoon Leader.SINCGARS.123456789".

A naming vocabulary will be required to assure that spelling and punctuation is consistent across all simulators. We feel that this vocabulary ought not be part of the standard, but rather a separate document administered by an appropriate joint-service panel. Of particular concern is the unified determination of weapon stations/pylons on aircraft as a large number of different configurations of stores can be supported. It will be necessary to communicate the standard numbering of stations to avoid inconsistent presentations between simulator visual systems.

<organization ID> - Identifies the entity's place in the field structure.

```
type Organizational_Unit_Descriptor is string(0..512);
```

The string would contain at least five hierarchical demarcations beginning as follows:

1) Country of deployment
2) Armed Forces branch of deployment
3) Corps, Fleet, or Command within the Branch
4) Division, Navy Squadron
5) Brigade, Regiment, Vessel, Wing
6) Battalion, Air Force Squadron
7) Company, Battery, Troop, Flight
8) Platoon, Element
9) Section
10) Squad
11) Individual

<sensor data> - Identifies the entity's sensor suite.

```
type Sensor_Definition is
  record
    Band : string(0..31);
    Gain : <number>;
      -- Units: dB
    Range : <32-bit number>;
      -- Units: BAM32+
    Operational : <number>;
      -- Units: operational % of capability
  end record;
```

<weapon data> - Identifies the entity's weapon suite. Note that full <weapon system> data is available on bursts of fire from a weapon when it is discharged.

```
type Weapon_Definition is
  record
    Caliber : <number>;
      -- Units: millimeters
    Ammo_Count : <number>;
    Ammo : array (1..Ammo_Count) of
      record
        Ammo_Name : String(0..12);
        Quantity : <number>;
      end record
  end record;
```

<fuel data> - Identifies fuel stores.

```
type Fuel_Class is (Gasoline, Diesel, Kerosene, JP4, Oil, Lube);
type Fuel_Definition is
  record
    Kind : Fuel_Class;
    Liters : <number>
  end record;
```

<articulation> - Identifies how the entity is articulated.

```
type Articulation_Class is (Rotary, Linear);
type Articulation_Definition is
  record
    Kind : Articulation_Class;
    DOF : <number>
    Parameter : array (1..DOF) of <number>;
  end record;
```

Numbers

In all these definitions, either 16-bit or 32-bit unsigned integers are used. These must in general be subtypes because Ada compilers are not required to support specific word sizes. In other implementation languages, there may be a more direct path to this definition.

```
type <number> is integer range 0..65535;
type <32-bit number> in integer range 0..4294967295;
```

Basic Encoding Rules

We support the position presented by Michael Sabo in his ISP position paper section 6.0 that the ASN.1 basic encoding rules should be used to translate the high-level structures presented here in Ada into bits for the lower protocol layers to transmit.

POSITION PAPER

Protocol Data Units for a Unified Simulation Internet

Reference: 131.rsc.209.23

Submitted by:
Michael Robkin
Randy Saunders
Hughes Simulation Systems, Inc.
1200 E. San Bernardino Road
West Covina, CA 91790

ABSTRACT

This paper describes the specific protocol data units (PDUs) applicable to a general-purpose simulation internet.

INTRODUCTION

The development of a standard for networking simulators must address a number of diverse topics, central topics among which are: (a) the architectural model used to define the network, (b) the communication model used to define how network entities interact, (c) the form and function of network interactions, and (d) the representation of network information in a standard format. This paper presents the form and function of messages that follow the architectural and communication models presented in our position paper reference 131.rsc.208.

The SIMNET PDUs developed by BBN Systems and Technologies have been offered as a potential baseline for such a standard. Two open conferences on the subject have surfaced a number of different issues bearing on a potential standard. With due appreciation to the above sources, this paper describes a different set of PDUs in order to generalize the simulation internet.

GENERIC FIELD TYPES

The fields in the PDUs presented in this paper are defined in a notation- and units-independent way to allow separate discussion of units and notation. The related position paper reference 437.mrc.100 provides our recommended notations and units. These PDUs provide the connectionless communication between simulators, and are intended as an application layer datagram protocol. The possibilities for connection-based extensions is discussed in our position paper reference 131.rsc.208, although we have no specific connection-based PDU recommendations at this time.

PDU Type

The following definition describes the overall structure of all PDUs. Data types used within other position papers are expressed inside angle brackets (e.g. <ID number>) to improve readability. As a result, the following fragments are not valid Ada unless the actual type names are inserted.

```
type Simulation_PDU(Kind : Simulation_PDU_Kind) is
  record
    Version : Simulation_Protocol_Version := Version_Feb_90;
      -- To distinguish this protocol from the SIMNET one
    PDU_Kind : Simulation_PDU_Kind := Kind;
      -- Identifies the type of PDU
    Exercise : Simulation_Exercise_ID := Current_Exercise;
      -- Identifies which exercise(s) the entity is in
    Entity : <ID number> := OwnShip_ID_Number;
      -- Identifies entity to which the PDU refers
    Source_Time : <time stamp>;
      -- Identifies when the PDU was created
    Receive_Time : <time stamp>;
      -- Identifies when this simulator received the PDU
    case Kind is
      when Identification_PDU =>
        Identification : Identification_Variant;
      when Configuration_PDU =>
        Configuration : Configuration_Variant;
      when Position_PDU =>
        Position : Position_Variant;
      when Explosion_PDU =>
        Explosion : Explosion_Variant;
      when Damage_PDU =>
        Damage : Damage_Variant;
      when Property_PDU =>
        Property : <property list>;
      when Error_PDU =>
        Error : Error_Variant;
      when Query_Identification_PDU =>
        Query_Identification : Query_Identification_Variant;
      when Query_Configuration_PDU =>
        Query_Configuration : Query_Identification_Variant;
      when Query_Position_PDU =>
        Query_Position : Query_Identification_Variant;
      when Query_All_Identification_PDU => null;
      when Query_All_Configuration_PDU => null;
      when Query_All_Position_PDU => null;
      when Query_Property_PDU =>
        Query_Property : Query_Property_Variant;
      when Query_General_Property_PDU =>
        Query_General_Property : Query_General_Property_Variant;
      when Query_General_Location_Property_PDU =>
        Query_Location_Property : Query_Location_Variant;
      when others => null;
    end case;
  end record;
```

Own State PDUs

Identification PDU. The following are non-changing attributes of an entity. This PDU contains the static qualities presented in the SIMNET VehicleAppearance and ActivateRequest PDUs. The rationale for eliminating the Activate PDUs is that the initialization of a system is not a generic problem, it should be handled via a connection-based extension. This PDU assumes that a given entity will not change its organization and marking characteristics during an exercise. This does not mean that a single simulator cannot change sides, markings, or crews, but only that it becomes a new network entity when it does so.

```
type Identification_Variant is
  record
    Entity_Kind : <entity type>;
      -- Gives network characteristics of the entity
    Entity_System : <weapon system>;
      -- Identifies the entity's weapon system
    Organizational_Element : <organization ID>;
      -- Hierarchical identification of the entity in terms
      -- of its role in this exercise
    Marking : Vehicle_Marking;
      -- Identifies the symbol set and marking of the entity
    Maximum_Velocity : <velocity>;
      -- Identifies the maximum velocity that can be attained
    Maximum_Firepower : <number>;
      -- Identifies the maximum number of exploding entities
      -- that this entity will create
  end record;
```

Configuration PDU. The following are specific characteristics about the entity that are type dependent and variable. It is important that static configuration parameters (e.g. gross weight of an M1 tank = 60,000 kg) need never be transmitted. If some simulator needs to know the weight of another vehicle that information can be determined from the <weapon system> data in the identification field. Alternatively, it could be queried so that the other simulator or a third party data server could provide the data.

```
type Configuration_Variant(Kind : <entity type>) is
  record
    case Kind is
      when Irrelevant_Class => null;
        -- No significant configuration data for artifacts
      when Static_Class =>
        Changed_Properties : <property list>;
        Sensor_Count : <number>;
        Sensor : array (1..Sensor_Count) of <sensor data>;
        -- Static entities may have <weapon system> dependent
        -- characteristics and sensors
```

```
when Platform_Class =>
  Changed_Properties : <property list>;
  Sensor_Count : <number>;
  Sensor : array (1..Sensor_Count) of <sensor data>;
  Weapon_Count : <number>;
  Weapon : array (1..Weapon_Count) of <weapon data>;
  -- Supports fixed gun emplacements
when Mobile_Platform_Class =>
  Changed_Properties : <property list>;
  Mobility_Status : <SIMNET MotiveSubsystems status data>;
  Fuel : <fuel data>;
  Sensor_Count : <number>;
  Sensor : array (1..Sensor_Count) of <sensor data>;
  Weapon_Count : <number>;
  Weapon : array (1..Weapon_Count) of <weapon data>;
  -- Supports mobile guns and similar platforms
when Articulated_Platform_Class =>
  Changed_Properties : <property list>;
  Mobility_Status : <SIMNET MotiveSubsystems status data>;
  Fuel : <fuel data>;
  Sensor_Count : <number>;
  Sensor : array (1..Sensor_Count) of <sensor data>;
  Turret_Count : <number>;
  Turret : array (1..Turret_Count) of
    record
      Turret_Status : <SIMNET TurretSubsystems status data>;
      Turret_to_Midline_Angle : <articulation>;
      Weapon_Count : <number>;
      Weapon : array (1..Weapon_Count) of <weapon data>;
    end record;
  Art_Count : <number>;
  Articulate : array (1..Art_Count) of <articulation>;
  -- Supports tanks, ships, and other complex vehicles
when Invisible_Observer_Class =>
  Changed_Properties : <property list>;
  Sensor_Count : <number>;
  Sensor : array (1..Sensor_Count) of <sensor data>;
  -- Supports "stealth" and Instructor vehicles
when Airborne_Class =>
  Changed_Properties : <property list>;
  Mobility_Status : <SIMNET MotiveSubsystems status data>;
  Fuel : <fuel data>;
  Sensor_Count : <number>;
  Sensor : array (1..Sensor_Count) of <sensor data>;
  -- Supports unarmed air vehicles
when Airborne_Platform_Class =>
  Changed_Properties : <property list>;
  Mobility_Status : <SIMNET MotiveSubsystems status data>;
  Fuel : <fuel data>;
  Sensor_Count : <number>;
  Sensor : array (1..Sensor_Count) of <sensor data>;
  Weapon_Count : <number>;
  Weapon : array (1..Weapon_Count) of <weapon data>;
  Art_Count : <number>;
  Articulate : array (1..Art_Count) of <articulation>;
  -- Supports normal air vehicles
```

```
    when Projectile_Class =>
      Changed_Properties : <property list>;
      Rounds_Count : <number>;
      Rounds_Rate : <number>;
      Tracer_Count : <number>;
      -- Supports rounds that follow ballistic rules.  See
      -- discussion of Explosion PDU for details
    when Missile_Class =>
      Changed_Properties : <property list>;
      Mobility_Status : <SIMNET MotiveSubsystems status data>;
      Fuel : <fuel data>;
      Rounds_Count : <number>;
      Rounds_Rate : <number>;
      Tracer_Count : <number>;
      Sensor_Count : <number>;
      Sensor : array (1..Sensor_Count) of <sensor data>;
      -- Supports and guidable munition or missile
    when Cloud_Class =>
      Changed_Properties : <property list>;
      Density : <number>;
        -- Units: 0..32767 partial fraction of atmosphere
      Mean_Height : <distance>;
      Mean_Width : <distance>;
      Mean_Length : <distance>;
      -- Supports clouds of smoke, chemical, and other agents
    when Flare_Class =>
      Changed_Properties : <property list>;
      Radiation_Level : <number>;
        -- Units: 0..65535 watts
      Burn_time : <number>;
        -- Units: 0..65535 seconds
      -- Supports point illumination sources.  Configuration
      -- PDUs are not sent when burn time is only changed
      -- parameter.
    when Emitter_Class =>
      Changed_Properties : <property list>;
      Frequency_Count : <number>;
      Frequency : array (1..Frequency_Count) of
        record
          Center : <32-bit number>;
            -- Units: Hertz
          Spectral_Width : <32-bit number>;
            -- Units: Hertz
          Radiation_Level : <number>;
            -- Units: 0..65535 watts
        end record;
    when Sensor_Class =>
      Changed_Properties : <property list>;
      Sensor_Count : <number>;
      Sensor : array (1..Sensor_Count) of <sensor data>;
      -- Supports moving sensors not on a platform
    when others => null;
  end case;
end record;
```

Position PDU. - The position PDU presents the dynamic characteristics of the entity. To provide for complexly articulated systems, this PDU contains only the angles involved. The remaining data is defined in the <articulation> part of the Configuration PDU.

```
type Position_Variant is
  record
    Latitude : <BAM32>;
    Longitude : <BAM32>;
    Radial_Range : <BAM32+>;
    Latitude_Velocity : <BAM32s per hour>;
    Longitude_Velocity : <BAM32s per hour>;
    Radial_Velocity : <BAM32+s per hour>;
    Pitch_Angle : <BAM16>;
    Roll_Angle : <BAM16>;
    Yaw_Angle : <BAM16>;
    Pitch_Rate : <BAM16s per second>;
    Roll_Rate : <BAM16s per second>;
    Yaw_Rate : <BAM16s per second>;
    Art_Count : <number>;
    Articulate : array (1..Art_Count) of <BAM16>;
  end record;
```

Explosion PDU. - The Explosion PDU defines the characteristics of explosive forces. It is possible to characterize an entire burst of fire in one PDU, but no provision is made for systematic gun slew. This allows short bursts of a high-rate weapon to be handled without flooding the network. We propose that no burst be longer than 30 seconds, but this figure is arbitrary and a doctrinally derived figure would be preferable.

```
type Explosion_Variant is
  record
    Latitude : <BAM32>;
    Longitude : <BAM32>;
    Radial_Range : <BAM32+>;
    Latitude_Velocity : <BAM32s per hour>;
    Longitude_Velocity : <BAM32s per hour>;
    Radial_Velocity : <BAM32+s per hour>;
    Pitch_Angle : <BAM16>;
    Roll_Angle : <BAM16>;
    Yaw_Angle : <BAM16>;
    Rounds_Count : <number>;
    Rounds_Rate : <number>;
    Tracer_Count : <number>;
    Round_Energy : <number>;
      -- Units: 0..655350 grams of explosive (10x)
    Directionality : <number>;
      -- Units: 0..65535 fraction of energy directed along
      -- the projectile's line of flight (65535 = 100%)
    Error_Radius : <BAM+>;
      -- Defines the circular mean error radius of the burst
  end record;
```

Damage PDU. - The Damage PDU is distinguished from a simple configuration PDU because it identifies a specific explosion as the source of the damage. This supports systems with otherwise unacceptable PDU loss rates by allowing regeneration of Explosion PDUs. It also provides enhanced analysis and after-action review by detailing how an entity has been damaged.

```
type Damage_Variant is
  record
    Damage_Cause : <ID number>;
    New_Configuration : Configuration_Variant;
  end record;
```

Error PDU. - The Error PDU is used to identify that a property is not supported by the sending entity.

```
type Error_Variant is
  record
    Property_Count : <number>
    Property : array (1..Property_Count) of
      record
        Name : string(1..32);
        Name_Length : integer range 1..32;
        Value : <32-bit number> := 0;
      end record;
  end record;
```

Property PDU. - The Property PDU is used to transmit the value of a property when there is no change associated with the property.

```
type <property list> is
  record
    Property_Count : <number>
    Property : array (1..Property_Count) of
      record
        Name : string(1..32);
        Name_Length : integer range 1..32;
        Value : <32-bit number>;
      end record;
  end record;
```

Defined Properties. We believe that a defined list of property names ought to be maintained outside the proposed MIL-STD. This would allow a protocol-independent means of introducing additional data to the simulation internet. When a new property is needed, a name not presently used would be assigned. All simulators would not need to be retrofitted, but would simply produce Error PDUs as defined above.

In a scenario where typical values provide adequate training, the requesting simulator could use a static value perhaps based on the <weapon system> or even <entity type>. If training would be impacted by the connection of two simulators that do not share a certain property set, a node on the network could be distinguished

as the arbiter of their characteristics. This distinguished node could respond to the Error PDU by producing an appropriate Property PDU. In this way, the network could be extended in phases and fielded simulators updated as resources become available, without impacting operations. These properties are a starting point.

Property Name	Value Meaning
Force Identifier	= SIMNET ForceID
Organizational Unit Array	= SIMNET OrganizationalUnit
Simulator Type	= SIMNET SimulatorType
Terrain Database	= SIMNET TerrainDatabaseID
Can Supply Ammunition	True(=1) or False(=0)
Can Supply Fuel	True(=1) or False(=0)
Can Recover Vehicles	True(=1) or False(=0)
Can Recover Crew	Quantity or False(=0)
Can Recover Injured	Quantity or False(=0)
Can Repair Vehicles	True(=1) or False(=0)
Can Repair Aircraft	True(=1) or False(=0)
Can Repair Vessels	True(=1) or False(=0)
Can Repair Submarines	True(=1) or False(=0)
Distinguished Representation	[TBD]
Odometer Reading	Kilometres
Engine Power	Watts
Battery Voltage	Millivolts
Vehicle Fire	= SIMNET appearance (0:2)
Vehicle Dust Cloud	Size of dust could in BAM32+
Vehicle Smoke Cloud	Size of dust cloud in BAM32+
Mean Temperature	Tenths of a degree C
Mean Noise Level	Bels
Mean Wind Speed	BAM32+ per hour
Wind Direction	BAM32

Query PDUs

Query operations could be handled as a connection-based extension to the protocols using the same PDUs simply by deleting the Target_ID fields and making that information part of the connection. Their inclusion is recommended because they provide a significant benefit when the use of intelligent network components is considered. In a system with intelligent agents there would be no need to relay a query all the way to the target simulator. In the minimalist case where no network intelligence is available it is equivalent to handle queries as connectionless or connection based. In this situation, we prefer the connectionless approach because it is more powerful in future applications.

Query Identification PDU, Query Configuration PDU, and Query Position PDU. All three of these PDUs request the designated entity to transmit a PDU when it would normally not do so because no information has changed.

```
type Query_Identification_Variant is
  record
    Target_ID : <ID number>
  end record;
```

Query All Identification PDU, Query All Configuration PDU, and Query All Position PDU. All three of these PDUs request all entities to provide the requested information. No information is specified.

Query Property PDU. The target is provided a list of properties of interest to the sender. If some properties are not supported, they are returned in an Error PDU. Supported properties are returned in a normal Property PDU.

```
type Query_Property_Variant is
  record
    Target_ID : <ID number>
    Property_Count : <number>
    Property : array (1..Property_Count) of
      record
        Name : string(1..32);
        Name_Length : integer range 1..32;
     end record;
  end record;
```

Query General Property PDU. General properties do not have a supported target; any entity may respond. Error PDUs are never produced. When an entity does not support a property, it remains silent

```
type Query_General_Property_Variant is
  record
    Property_Count : <number>
    Property : array (1..Property_Count) of
      record
        Name : string(1..32);
        Name_Length : integer range 1..32;
     end record;
  end record;
```

Query General Location Property PDU. Location properties do not have a supported target; any entity may respond. Error PDUs are never produced. When an entity does not support a property, it remains silent.

```
type Query_Location_Variant is
  record
    Latitude : <BAM32>;
    Longitude : <BAM32>;
    Radial_Range : <BAM32+>;
    Range_Error_Acceptable : <BAM32+>;
    Property_Count : <number>
    Property : array (1..Property_Count) of
      record
        Name : string(1..32);
        Name_Length : integer range 1..32;
     end record;
  end record;
```

CONCLUSION

The connectionless PDU scheme originated in SIMNET provides a good concept for network interface design. The proposed changes in details follow the principles started in SIMNET while generalizing to a network of broader application.